M. FLOROVICI
V. MARINOIU
I. DUMITRESCU
I. MACRI
I. LOCOVEI

APLICAȚII INGINEREȘTI ALE CALCULATOARELOR

OPTIMIZĂRI

Scanned with OKEN Scanner

MINISTERUL EDUCAȚIEI ȘI ÎNVĂȚĂMÎNTULUI

Conf. dr. ing. ION I. DUMITRESCU

Ing. MIHAI FLOROVICI
Asist. ing. ION MACRI

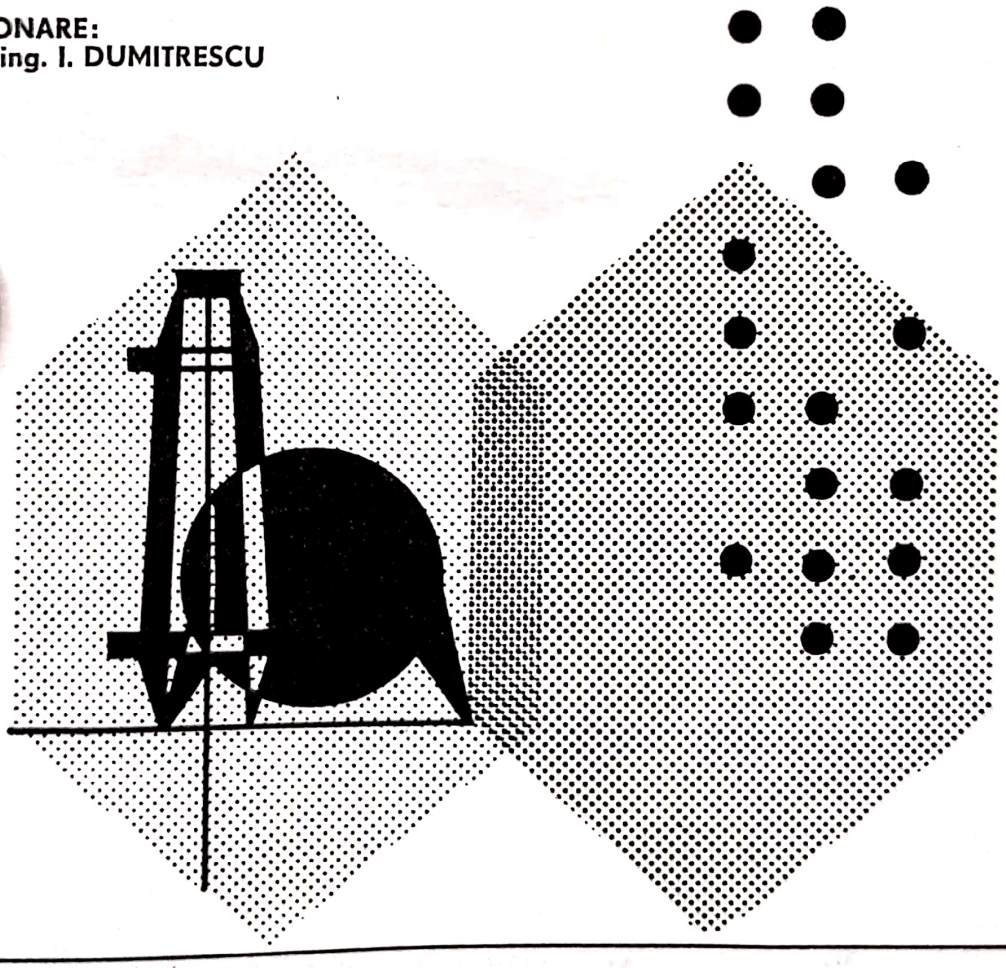
Șef lucrări dr. ing. VASILE MARINOIU
Mat. ION LOCOVEI

BIBLIOTECA PERSONALĂ
NR.
Mihaela și Viorel Lazăr

APLICAȚII INGINEREȘTI ALE CALCULATOARELOR

OPTIMIZĂRI

COORDONARE:
conf. dr. ing. I. DUMITRESCU



EDITURA DIDACTICĂ ȘI PEDAGOGICĂ, BUCUREȘTI

Scanned with OKEN Scanner

Cuprinsul volumului 1

APLICAȚII INGINEREȘTI ALE CALCULATOARELOR CU EXEMPLE DIN INDUSTRIA PETROLIERĂ ȘI PETROCHIMICĂ

Capitolul 1. PRELUCRAREA AUTOMATĂ A DATELOR (43 pagini)

1.1. Configurația sistemelor de calcul automat
1.2. Reprezentarea informației în echipamentele automate de calcul
1.3. Principiile prelucrării automate a datelor
1.4. Limbaje algoritmice de programare
1.5. Sisteme de operare

Capitolul 2. BAZELE PROGRAMĂRII ÎN FORTRAN (82 pagini)

2.1. Structura programelor în FORTRAN
2.2. Elementele de bază ale limbajului FORTRAN
2.3. Instrucțiuni de atribuire
2.4. Instrucțiuni de salt
2.5. Instrucțiuni de ciclare
2.6. Instrucțiuni de intrare/ieșire
2.7. Instrucțiuni proprii funcțiilor și subprogramelor
2.8. Directive de comandă
2.9. Fișiere utilizabile în FORTRAN

Capitolul 3. INIȚIERE ÎN PL/1 (49 pagini)

3.1. Elementele unui program PL/1
3.2. Date
3.3. Expresii
3.4. Instrucțiuni
3.5. Instrucțiuni de intrare/ieșire pentru fișierele standard
3.6. Blocuri și activarea blocurilor
3.7. Alocarea memoriei
3.8. Recunoașterea numelor
3.9. Subrutine și funcții
3.10. Atributul PICTURE
3.11. Instrucțiunea ON
3.12. Fișiere pe discuri și benzi magnetice
3.13. Exemplu de program PL/1

Scanned with OKEN Scanner

Capitolul 4. METODE NUMERICE (216 pagini)

4.1. Bazele metodelor numerice
4.2. Prelucrarea datelor experimentale
4.3. Rezolvarea numerică a ecuaţiilor neliniare
4.4. Rezolvarea numerică a sistemelor de ecuaţii algebrice liniare
4.5. Rezolvarea numerică a ecuaţiilor diferenţiale ordinare
4.6. Noţiuni cu privire la rezolvarea numerică a problemelor cu condiţii la limită în care intervin ecuaţii cu derivate parţiale

Capitolul 5. APLICAŢII (40 pagini)

5.1. Studiul proceselor de filtraţie staţionare din zăcămintele petrolifere
5.2. Sinteza profilului organelor de obturare ale robinetelor de reglare
5.3. Determinarea eforturilor din elementele turlelor de foraj de tip „A"
5.4. Calculul compoziţiei chimice a produselor rezultate din reacţia de piroliză a etanului

Sumarul volumului 3

APLICAŢII INGINEREŞTI ALE CALCULATOARELOR
— CALCULATOARE DE PROCES

Capitolul 10. UTILIZAREA SISTEMELOR DE CALCUL LA CONDUCEREA PROCESELOR INDUSTRIALE

Capitolul 11. SISTEME DE ECHIPAMENTE PENTRU CALCULATOARELE DE PROCES

Capitolul 12. SISTEME DE PROGRAME PENTRU CALCULATOARELE DE PROCES

Capitolul 13. EXEMPLE DE APLICAŢII ALE CALCULATOARELOR DE PROCES:
13.1. în industria extractivă a petrolului şi gazelor naturale
13.2. în industria metalurgică
13.3. în industria constructoare de maşini
13.4. în industria petrochimică şi în rafinăriile de petrol

Scanned with OKEN Scanner

PREFAȚĂ

Interesul cu care a fost primită cartea **„Aplicații inginerești ale calculatoarelor** *cu exemple din industria petrolieră și petrochimică", dar mai ales observația cititorilor în legătură cu faptul că utilizarea de către ingineri a sistemelor de calcul nu poate fi limitată numai la aplicațiile în tehnică ale metodelor numerice, ne-au determinat să elaborăm alte două volume, cu același titlu, care să continue lucrarea amintită, completînd-o cu alte aplicații specifice domeniului ingineresc. Aceste două noi volume ale lucrării „Aplicații inginerești ale calculatoarelor", subintitulate:* **optimizări** și **calculatoare de proces,** *sînt menite să asigure materialul de bază pentru studiul mai multor discipline predate la cele trei facultăți ale Institutului de petrol și gaze Ploiești și anume: „Matematici II (Programare)" la Utilaj, „Programare și calculatoare" la Foraj, „Programare" la Tehnologie, „Utilizarea calculatoarelor" la Utilaj chimic și petrochimic, toate cursurile de automatizare a proceselor industriale, cursurile postuniversitare de prelucrare automată a datelor (pentru modulele: „Aplicații programare", „Biblioteca pentru programare liniară" și „Aplicații tehnico-științifice"), precum și pentru „Aplicațiile calculatoarelor" (de la cursul internațional UNESCO). Ca urmare, aceste noi volume au fost concepute pentru a fi utile, în primul rînd, studenților Institutului de petrol și gaze, precum și participanților la cursurile postuniversitare din acest institut, dar s-a avut în vedere și posibilitatea consultării lor de către numeroșii ingineri petroliști (și de alte specialități), care se ocupă cu aplicațiile tehnicii de calcul în industrie, cercetare și proiectare.*

Prezentul volum, al doilea al lucrării „Aplicații inginerești ale calculatoarelor", are ca temă **optimizarea proceselor industriale,** *adică unul din domeniile cele mai fertile de aplicații ale calculatoarelor, domeniu în care utilizarea sistemelor de calcul automat se realizează cu maximă eficiență tehnică și economică.*

După cum se știe, activitatea în domeniul producției industriale și a conducerii întreprinderilor a devenit în prezent atît de complexă (ca să nu spunem complicată!) încît alegerea soluției „celei mai bune" prin metodele matematicii clasice, bazate pe aplicațiile calculului diferențial și ale calculului variațional (problemele de extrem din așa-numita matematică tradițională), nu mai sînt suficiente. Astăzi, problema alegerii „celei mai bune" variante în domeniul producției și conducerii, care apare frecvent în activitatea industrială curentă, este denumită o problemă de optimizare. *Modelarea (adică descrierea matematică) a problemelor de optimizare permite găsirea* soluțiilor optime, *adică a celor mai bune variante, apreciate ca atare din anumite puncte*

Scanned with OKEN Scanner

de vedere prin diferite criterii legate de: calitatea produselor, volumul producției, utilizarea capacităților de producție, timpul de realizare a producției, cheltuielile de producție, beneficiul ș.a.m.d., care— după caz — se cer a fi maxime (de exemplu: volumul producției, beneficiul etc.) sau minime (timpul de realizare a producției, cheltuielile etc.).

Referindu-ne la conducerea unui proces industrial în care intervin mai multe variabile de stare, obiectivul urmărit este — în principiu — acela de a realiza în mod economic o performanță superioară a procesului. Un sistem ideal de conducere (de tip „hard-soft") va ține seama de toate interacțiunile semnificative dintre variabilele procesului, precum și de influențele lor economice reciproce, luînd acea decizie de conducere care va avea ca efect obținerea performanței optime a procesului. Realizarea conducerii optimale este legată direct și de un alt factor — acela al efectului perturbațiilor (de altfel, dacă nu ar exista perturbații nici nu ar fi necesară vreo acțiune de conducere). În lipsa acestor perturbații, orice proces s-ar desfășura în condiții optime fără nici o intervenție: atunci cînd apare însă o perturbație trebuie să se realizeze anumite schimbări în proces pentru a compensa efectul perturbației.

Modelele (mai precis modelele matematice) folosite pentru conducerea optimală a proceselor industriale pot fi cu sau fără restricții. Așa-zisa optimizare fără restricții *se poate face determinînd punctul de funcționare optim (maxim/minim), prin rezolvarea problemei clasice de extrem, care poate fi efectuată analitic prin calcule elementare și metode numerice (prezentate de noi în primul volum al acestei lucrări* [7]*). Procesele industriale sînt supuse, cel mai adesea, la o serie de restricții de care trebuie să se țină seama la alegerea variantei optime (variabilele modelului trebuie să respecte condiții restrictive din motive fizice, tehnice, de securitate, sau rezultînd din strategia conducerii întreprinderilor). Astfel, cel mai adesea, conducerea proceselor se realizează prin optimizarea cu restricții.*

În cadrul acestui volum, purtînd subtitlul **optimizări,** *ne vom ocupa cu prezentarea optimizării cu restricții. Ca urmare, cartea a fost organizată în funcție de modelele utilizate în cadrul optimizării cu restricții și anume:*

— programarea liniară *(capitolul 6), referitoare la metodele de calcul pentru rezolvarea problemelor de optimizare cu restricții în care atît funcția obiectiv cît și restricțiile sînt reprezentate prin expresii liniare. Cele mai multe cazuri practice de optimizare pot fi tratate prin acest model, a cărui rezolvare face apel la metodele algebrei liniare;*

— programarea neliniară *(capitolul 7) care, referindu-se la sistemele neliniare, operează cu modele neliniare (de exemplu, în numeroase cazuri, funcția obiectiv are o formă pătratică). Rezolvarea problemelor de programare neliniară se face prin metodele gradientului și de căutare (directă, prin sondaj, prin metoda rețelei Monte-Carlo etc.);*

— programarea dinamică *(capitolul 8), care este o metodă eficientă de optimizare în cazul conducerii staționar-optimale a proceselor, dacă problema poate fi formulată ca o problemă de decizie cu mai multe etape (decizii secvențiale), ca de exemplu cazul unui grup de utilaje care din punct de vedere tehnologic lucrează în serie. Fiecare utilaj al liniei tehnologice-serie poate fi considerat ca o etapă, deciziile putînd fi luate separat, pentru fiecare etapă (utilaj); problema se reduce deci la a determina o astfel de „secvență" a acțiunilor de conducere încît funcționarea tuturor etapelor în ansamblu (deci*

Scanned with OKEN Scanner

a liniei tehnologice) să fie optimă. Dacă etapele procesului sînt formulate ca intervale de timp (și nu ca intervale de spațiu sau ca unități de utilaj tehnologic), atunci programarea dinamică poate fi utilizată pentru a obține secvența optimă în timp a deciziilor (adică „drumul" dinamic optim).

Ultimul capitol al acestui volum este conscarat unor metode de optimizare bazate pe aplicațiile calculului probabilităților și ale statisticii matematice. Dintre numeroasele metode statistice ale cercetării operaţionale *am ales numai cîteva și anume acelea care au o mai mare utilizare în industrie și tehnică (aplicațiile teoriei firelor de așteptare, ale teoriei stocurilor și ale teoriei echipamentului).*

Ca și primul volum, lucrarea de față — în afara intențiilor didactice — își propune și scopuri practice-aplicative. În consecință, ea a fost elaborată în așa fel încît să poată fi direct utilizată, ca un îndrumar, la rezolvarea unor probleme concrete ce apar uzual în practica conducerii proceselor și întreprinderilor industriale. Ca urmare, în fiecare capitol, după formularea succintă a metodei și precizarea domeniului său de aplicativitate, sînt prezentați principalii algoritmi și procedee practice bazate pe utilizarea sistemelor de calcul automat și a pachetelor de programare aflate în uzul centrelor de calcul, aplicarea efectivă a fiecărui procedeu fiind ilustrată printr-un exemplu concret, tratat „in extenso" (cu inserarea și a programului sursă utilizat, redactat în limbajul FORTRAN).

În acest fel, am căutat să răspundem unor exigențe impuse de necesitatea continuei ridicări a eficienței producției în toate ramurile industriale, pe baza folosirii optime a resurselor prin aplicarea celor mai moderne metode de organizare și conducere. În acest sens, nu putem să încheiem această prefață fără a reaminti cititorului obligația pe care o are ca inginer de a contribui la realizarea cerinței ca cincinalul 1976—1980 „... să devină cincinalul revoluției tehnico-științifice, al afirmării largi a cuceririlor celor mai avansate ale cunoașterii în toate ramurile și sectoarele industriei românești."), precum și la înfăptuirea programului de modernizare și ridicare a nivelului tehnic și calitativ al întregii noastre economii, prevăzut în „Directivele Congresului al XI-lea al Partidului Comunist Român cu privire la planul cincinal 1976-1980 și liniile directoare ale dezvoltării economico-sociale a României pentru perioada 1981—1990".*

Conf. dr. ing. I. DUMITRESCU, prof. dr. ing. STELIAN DUMITRESCU

Ploiești,
iulie 1975

* Ceaușescu, N. *Raport la cel de al XI-lea Congres al Partidului Comunist Român*. București, Editura politică, 1974 (p. 51).

Scanned with OKEN Scanner

CUPRINS

Prefață 5

Capitolul 6. APLICAȚIILE PROGRAMĂRII LINIARE 11

6.1. Modelarea prin programe liniare de optimizare 12
6.2. Metode de rezolvare a programelor liniare 26
Generalități 26
Metoda simplex 30
Metode de rezolvare a problemelor de transport 56
6.3. Pachete de programe specializate pentru rezolvarea problemelor de programare liniară 74
Program pentru problemele de transport — TRANSPOR 75
Programul OPALINE 83

Capitolul 7. APLICAȚIILE PROGRAMĂRII NELINIARE 105

7.1. Programe neliniare 105
7.2. Algoritmi pentru programele neliniare 112
Algoritmul liniarizării secționale 112
O nouă variantă a programării liniare secționale 119
Algoritmi gradient 125
Algoritmul simplex 127
Algoritmul explorării 128
7.3. Un exemplu de aplicație — optimizarea procesului de piroliză a etanului 133

Capitolul 8. APLICAȚIILE PROGRAMĂRII DINAMICE 145

8.1. Introducere în programarea dinamică 146
8.2. Exemple de aplicații 150
8.3. Principiile modelării dinamice 172
8.4. Modelarea dinamică prin metoda lui Forrester 177
8.5. Un exemplu de aplicație al metodei Forrester 197
8.6. Algoritm de conducere optimală 203

Capitolul 9. OPTIMIZĂRI PRIN METODE STATISTICE ALE CERCETĂRII OPERAȚIONALE 212

9.1. Aplicațiile firelor de așteptare 212
Terminologie 212
Clasificarea modelelor de așteptare 215
*Modele de așteptare ale clasei PPS** 217
Un exemplu de aplicație 226

Scanned with OKEN Scanner

9.2. Aplicaţiile teoriei stocurilor 244
Elementele de bază ale teoriei stocurilor 244
Modele probabilistice ale teoriei stocurilor 248
Exemple 254
9.3. Aplicaţiile teoriei echipamentului 273
Probabilităţi şi curbe de supravieţuire 274
Un exemplu 277
Bibliografie 287

Scanned with OKEN Scanner

Capitolul 6

APLICAŢIILE PROGRAMĂRII LINIARE

Numeroase probleme de optimizare cu restricţii pot fi rezolvate prin utilizarea unor modele de tipul programelor liniare. Programarea liniară poate fi aplicată în cazul în care procesele din cadrul sistemului analizat pot fi descrise, din punct de vedere matematic, prin relaţii liniare de forma :

$$a_1x_1 + a_2x_2 + \ldots + a_ix_i + \ldots + a_nx_n = a_{n+1},$$

în care a_i ($i = 1, 2, \ldots, n + 1$) sînt coeficienţi constanţi, iar x_i ($i = 1, 2, \ldots n$) sînt variabile independente ce trebuie determinate în aşa fel încît să se obţină o stare optimă a sistemului.

După cum se ştie, orice problemă de programare liniară constă în a determina situaţia în care un anumit *scop*, aprioric stabilit în funcţie de natura proceselor analizate, poate fi realizat în condiţii optime. Expresia matematică care modelează acest scop poartă denumirea de *funcţie obiectiv* sau *funcţie criteriu* (deseori sînt utilizate şi denumirile : *funcţie scop*, *funcţie de eficienţă*, *funcţie de optimizare* sau, uneori, *funcţie economică*).

Funcţia obiectiv trebuie să satisfacă două condiţii : să fie liniară şi unică. Unicitatea funcţiei obiectiv rezultă din faptul că unui model care descrie un anumit sistem nu i se poate asocia decît numai un singur criteriu de optimizare. Astfel, activitatea unei întreprinderi nu poate fi optimizată simultan din mai multe puncte de vedere ; de exemplu, dacă s-ar urmări obţinerea unei valori maxime a producţiei, rezultatul ar fi diferit de cel ce ar rezulta în cazul în care s-ar dori realizarea unui beneficiu maxim (sau structura producţiei obţinute prin „maximizarea" beneficiului ar fi diferită de structura ce ar rezulta luînd ca criteriu de optimizare reducerea la minimum posibil a cheltuielilor de producţie).

Situaţia specifică a unui sistem dat, a cărui stare este apreciată prin diverşi parametri fizici, tehnici şi economici, determină o serie de condiţii pe care trebuie să le respecte aceşti parametri, ce pot fi modelate — adică descrise într-o formă matematică — prin relaţii de condiţie, denumite *restricţii*. Aceste restricţii, care sînt exprimate analitic prin relaţii de egalitate sau inegalitate, trebuie să fie şi ele liniare.

Funcţia obiectiv şi sistemul de restricţii alcătuiesc modelul procesului ce se cere a fi optimizat, model denumit *program liniar de optimizare*.

Scanned with OKEN Scanner

În cadrul acestui capitol, presupunînd că cititorul are pregătirea matematică şi de informatică necesară, vom prezenta: procedeul modelării prin programe liniare, principalele metode pentru rezolvarea problemelor de programare liniară, tehnici de utilizare a sistemelor de calcul automat în programarea liniară şi pachete de programe uzuale în acest domeniu.

Pentru a ilustra modul cum se aplică metodele prezentate, în toate cazurile am făcut apel la exemple concrete, tipice, care să înlesnească realizarea de către cititor a oricărei alte aplicaţii.

6.1. MODELAREA PRIN PROGRAME LINIARE DE OPTIMIZARE

În forma ei generală, o problemă de optimizare prin programare liniară se formulează astfel:

Să se determine valoarea maximă/minimă a funcţiei liniare:

$$F = c_1x_1 + c_2x_2 + \ldots + c_nx_n, \tag{6—1}$$

astfel încît variabilele nenegative x_i $(i = 1, 2, \ldots, n)$ să satisfacă următoarele inegalităţi liniare:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leqslant b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leqslant b_2 \\ \quad\vdots \qquad\quad \vdots \qquad\qquad\quad \vdots \quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leqslant b_n \end{cases} \tag{6—2}$$

unde coeficienţi $a_{ij}(i = 1, 2, \ldots, m; j = 1, 2, \ldots, n)$ şi $b_i(i = 1, 2, \ldots, m)$ sînt numere reale în legătură cu care nu vom introduce nici o ipoteză particulară.

Necunoscutele $x_i(i = 1, 2, \ldots, n)$, care reprezintă variabilele de stare ale sistemului economic, industrial sau fizic analizat, nu pot avea sens — de cele mai multe ori — decît dacă sînt pozitive, ceea ce se exprimă prin condiţiile de nenegativitate:

$$\begin{cases} x_1 \geqslant 0 \\ x_2 \geqslant 0 \\ \quad\vdots \\ x_n \geqslant 0 \end{cases} \tag{6—3}$$

În modelul de mai sus, funcţia (6—1) este funcţia obiectiv, iar sistemul de inegalităţi liniare (6—2) reprezintă restricţiile.

Este evident că minimizarea funcţiei F se poate reduce, ca formalizare matematică, la problema determinării maximului aceleiaşi funcţii F. Într-adevăr, între maximul funcţiei $(max\ F)$ şi minimul aceleiaşi funcţii $(min\ F)$ există relaţia:

$$min\ F = -max\ (-F),$$

Scanned with OKEN Scanner

deoarece dacă înmulţim ambii membri ai egalităţi (6—1) cu -1, rezultă:

$$-F = -c_1x_1 + (-c_2)x_2 + \ldots + (-c_n)x_n$$

şi deci $max\,(-F) = -min\,F$.

Adăugînd la cele de mai sus şi observaţia că sistemul de inegalităţi (6—2) poate fi transformat într-un sistem de egalităţi prin ataşarea unor aşa-numite *variabile de ecart* $x_{n+i} \geqslant 0$ $(i = 1, 2, \ldots, m)$, înţelegem de ce modelul de programare liniară (6—1), (6—2), (6—3) poate fi prezentat, dacă utilizăm şi scrierea condensată, în următoarea formă:

$$F = \sum_{i=1}^{n} c_i x_i + \sum_{i=1}^{m} 0 \cdot x_{n+i} = max \qquad (6—1a)$$

$$\left(\sum_{j=1}^{n} a_{ij}x_j\right) + x_{n+i} = b_i \quad i = 1, 2, \ldots, m, \qquad (6—2a)$$

$$x_i \geqslant 0 \qquad i = 1, 2, \ldots, n + m. \qquad (6—3a)$$

Forma (6—1), (6—2), (6—3) se numeşte *forma canonică* a unui program liniar, iar (6—1*a*), (6—2*a*), (6—3*a*) este numită *forma standard* a modelului de programare liniară.

Evident că programul liniar, în ambele forme de mai sus, poate fi scris mai concis şi în mod matriceal. Astfel, în cazul formei standard a programului liniar, dacă folosim notaţiile:

$\mathbf{C} = (c_1\ c_2\ \ldots\ c_n)$ pentru vectorul linie al coeficienţilor funcţiei obiectiv,

$\mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$ pentru vectorul coloană al termenilor liberi din sistemul restricţiilor,

$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ pentru vectorul coloană al variabilelor,

$\mathbf{E} = \begin{bmatrix} x_{n+1} \\ x_{n+2} \\ \vdots \\ x_{n+m} \end{bmatrix}$ pentru vectorul coloană al variabilelor de ecart,

şi

$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix}$ pentru matricea coeficienţilor din sistemul re stricţiilor,

Scanned with OKEN Scanner

modelul (6—1*a*), (6—2*a*), (6—3*a*) poate fi scris şi în următoarea formă matriceală :

$$F = \mathbf{CX} + \mathbf{OE}^t = max, \qquad (6-1b)$$

$$\mathbf{AX} + \mathbf{IE} = \mathbf{B}, \qquad (6-2b)$$

$$\mathbf{X} \geqslant \mathbf{0}, \quad \mathbf{E} \geqslant \mathbf{0}, \qquad (6-3b)$$

unde $\mathbf{E}^t$ este matricea ecart transpusă, iar $\mathbf{I}$ — matricea unitate.

În funcţie de specificul claselor de probleme care se optimizează, modelele de programare liniară tipice au denumiri împrumutate de la problemele pe care le rezolvă şi anume : problema transportului → program de transport, problema dietei sau a amestecurilor → program de amestec, problema investiţiilor, problema încărcării maşinilor, problema debitării materialelor sau a croirii, problema investiţiilor, problema ordonanţării ş.a.m.d.

Exemple

Ex. 6—1. Pentru început, vom considera un exemplu foarte simplu.

La unul din bancurile de lucru ale atelierului de reparaţii dintr-o schelă de extracţie se pot face două categorii de reparaţii. Activitatea desfăşurată la acest banc de lucru poate fi analizată, din punct de vedere tehnic şi economic, prin intermediul următorilor parametri şi indicatori :

— consumul unitar de materiale,
— consumul unitar de forţă de muncă,
— fondul de timp zilnic,
— valoarea reparaţiilor efectuate,
— structura volumului de reparaţii etc.

Materialele şi forţa de muncă reprezintă *resursele* de care dispunem pentru efectuarea reparaţiilor.

Notînd cu x_1 şi x_2 volumul (în „bucăţi") al celor două categorii de reparaţii ce se pot efectua la acest banc de lucru şi considerînd că valorile parametrilor de mai sus sînt cele indicate în tabela 6—1, modelul pentru

Tabela 6—1

Felul repartiţiei / Resurse	x_1	x_2	Total resurse
Materiale (ore)	3	5	15
Forţă de muncă (ore)	6	2	24
Preţ unitar de reparaţii (lei/buc)	200	100	—

optimizarea activităţii la locul de muncă analizat poate considera ca scop stabilirea structurii volumului de reparaţii (x_1 şi x_2) astfel încît valoarea lui să fie maximă.

Scanned with OKEN Scanner

Atunci, modelul de optimizare ce rezultă pe baza datelor din tabela 6—1 este următorul:

$$F = 200\,x_1 + 100\,x_2 = max \qquad (6-4)$$

$$\begin{cases} 3x_1 + 5x_2 \leqslant 15 \\ 6x_1 + 2x_2 \leqslant 24 \end{cases} \qquad (6-5)$$

şi $x_1 \geqslant 0$, $x_2 \geqslant 0$.

Relaţia (6—4) exprimă funcţia obiectiv (rezultă direct pe baza datelor de pe ultima linie a tabelei 6—1), iar sistemul (6—5) reprezintă restricţiile.

Această problemă constituie un exemplu simplu de *program liniar pentru folosirea eficientă a resurselor limitate.*

Ex. 6—2 [21]. Un trust de extracţie a ţiţeiului are posibilitatea desfăşurării activităţii sale pe cinci structuri petroliere diferite (pe care să le notăm prin I, II, III, IV şi V), în care — conform programului de exploatare — se poate săpa un anumit număr maxim de sonde.

Conform planului de investiţii, trustul dispune de un anumit fond pentru lucrări de investiţii (pentru săparea de noi sonde), să zicem în valoarea de 300.000.000 lei/an.

Pe de altă parte, trustul — conform planului de producţie — are anumite sarcini de producţie, pe care trebuie să le realizeze prin intermediul sondelor noi săpate; astfel, să presupunem că producţia planificată din sondele noi este de 700.000 tone/an.

Problema care se pune (o problemă tipică de investiţii) este aceea a determinării numărului x_1, x_2, x_3, x_4 şi x_5 de sonde ce trebuie săpate în fiecare structură astfel ca trustul să realizeze o utilizare cît mai raţională şi eficientă a investiţiilor.

Aceasta se poate realiza luînd în considerare următorii parametri:

— numărul maxim al sondelor ce se pot săpa, într-un an, în fiecare structură;

— producţia medie a sondelor noi săpate, pe fiecare structură;

— investiţiile unitare (pe sonda nou săpată);

— cheltuielile anuale de exploatare, pe sonde.

În scopul unei mai bune ilustrări a acestui exemplu, în tabela 6—2 considerăm anumite valori numerice pentru parametrii indicaţi mai sus.

Tabela 6—2

Structura	Numărul maxim de sonde	Producţia medie pe sonda nouă (mii tone/an)	Investiţiile pe sonda nouă milioane lei/an)	Cheltuieli de exploatare pe sondă (milioane lei/an)
I	70	5	2	0,4
II	50	6	2,5	0,43
III	40	6,5	3	0,46
IV	30	7	3,5	0,48
V	25	8	4,2	0,52

Avînd în vedere datele din tabela 6—2, se pot alege mai multe variante în ceea ce priveşte criteriul de optimizare. Astfel de criterii pot fi:

a) minimizarea cheltuielilor de exploatare;

b) minimizarea investiţiilor;

Scanned with OKEN Scanner

c) maximizarea producţiei (în primul an de exploatare a sondelor noi sau pe o perioadă care să ţină seama şi de declinul sondelor).

După cum am mai arătat, pentru obţinerea unei soluţii optime trebuie să alegem numai un singur criteriu de optimizare. Astfel, modelele de optimizare vor fi:

— *în cazul criteriului a)*, referitor la minimizarea cheltuielilor de exploatare a sondelor noi:

$$F = 0{,}4x_1 + 0{,}43x_2 + 0{,}46x_3 + 0{,}48x_4 + 0{,}52x_5 = min, \qquad (6-6)$$

$$\begin{cases} 5x_1 + 6\ x_2 + 6{,}5x_3 + \ 7x_4 + \ 8x_5 \geqslant 700 & (6-7/1) \\ 2x_1 + 2{,}5x_2 + \ 3x_3 + 3{,}5x_4 + 4{,}2x_5 \leqslant 300 & (6-7/2) \\ x_1 \leqslant 70 & (6-7/3) \\ x_2 \leqslant 50 & (6-7/4) \\ x_3 \leqslant 40 & (6-7/5) \\ x_4 \leqslant 30 & (6-7/6) \\ x_5 \leqslant 25 & (6-7/7) \end{cases}$$

unde x_i $(i = 1, 2, \ldots, 5)$ reprezintă numărul sondelor ce vor trebui săpate în cele 5 structuri (x_1 sonde în structura I, x_2 sonde în structura II, ..., x_5 sonde în structura V).

Evident, la relaţiile (6—6) şi (6—7) trebuie ataşate şi condiţiile de nenegativitate:

$$x_i \geqslant 0 \qquad i = 1, 2, \ldots, 5.$$

Relaţia (6—6), care reprezintă funcţia obiectiv, rezultă din ultima coloană a tabelei 6—2. Restricţiile se obţin astfel: (6—7/1) din coloana a treia (restricţia de producţie), (6—7/2) din coloana a patra (restricţia de investiţii), iar (6—7/3), ..., (6—7/7) din coloana a doua (restricţiile relative la numărul maxim de sonde ce pot fi săpate);

— *în cazul criteriului b)*, referitor la minimizarea investiţiilor necesare pentru săparea noilor sonde:

$$F = 2x_1 + 2{,}5x_2 + 3x_3 + 3{,}5x_4 + 4{,}2x_5 = min, \qquad (6-8)$$

$$\begin{cases} 5x_1 + 6x_2 + 6{,}5x_3 + 7x_4 + 8x_5 \geqslant 700 & (6-9/1) \\ x_1 \leqslant 70 & (6-9/2) \\ x_2 \leqslant 50 & (6-9/3) \\ x_3 \leqslant 40 & (6-9/4) \\ x_4 \leqslant 30 & (6-9/5) \\ x_5 \leqslant 25 & (6-9/6) \end{cases}$$

şi condiţiile de nenegativitate:

$$x_i \geqslant 0 \qquad i = 1, 2, \ldots, 5.$$

Relaţia (6—8) este funcţia obiectiv (scrierea ei se face pe baza datelor din coloana a patra a tabelei 6—2),iar sistemul (6—9) reprezintă restricţiile (6—9/1 se scrie cu datele de producţie din coloana a treia şi 6—9/2, ... 6—9/6 se scriu cu datele din coloana a doua);

Scanned with OKEN Scanner

— *în cazul c)*, referitor la optimizarea producţiei în primul an de exploatare a noilor sonde:

$$F = 5x_1 + 6x_2 + 6{,}5x_3 + 7x_4 + 8x_5 = max, \qquad (6-10)$$

$$\begin{cases} 2x_1 + 2{,}5x_2 + 3x_3 + 3{,}5x_4 + 4{,}2x_5 \leqslant 300 & (6-11/1) \\ x_1 \leqslant 70 & (6-11/2) \\ x_2 \leqslant 50 & (6-11/3) \\ x_3 \leqslant 40 & (6-11/4) \\ x_4 \leqslant 30 & (6-11/5) \\ x_5 \leqslant 25 & (6-11/6) \end{cases}$$

şi condiţiile de nenegativitate:

$$x_i \geqslant 0 \qquad i = 1, 2, \ldots, 5.$$

Funcţia (6—10) este funcţia obiectiv (se scrie cu datele din coloana patru — tabela 6—2), iar (6—11) este sistemul restricţiilor.

În toate cazurile de mai sus, se mai introduce condiţia ca $x_i \in N$ ($i = 1, 2, \ldots, 5$), unde N este mulţimea numerelor naturale, deoarece numărul sondelor care se sapă nu poate fi decît un număr întreg pozitiv sau zero (nu se sapă nici o sondă). Astfel, problemele *a)*, *b)* şi *c)* sînt probleme de programare liniară în numere întregi.

Ex. 6—3. Este cunoscut faptul că diversele sortimente de produse petroliere, aşa cum sînt puse în vînzare, se obţin prin operaţii de amestec, în diferite proporţii, a constituenţilor petrolieri de bază (livraţi de rafinării), astfel încît să se obţină indicii calitativi prescrişi pentru fiecare sortiment în parte.

Operaţia de amestec a componenţilor de bază se poate realiza în numeroase moduri (variante), astfel încît s-a ajuns şi în acest caz la necesitatea obţinerii unor *reţete optime de amestec*. Optimizarea se realizează pe baza unor criterii tehnico-economice, urmărindu-se ca în funcţie de diversele calităţi ale componenţilor să se determine o astfel de proporţie a acestora încît să se obţină sortimente comerciale care la preţ de cost minim (sau beneficiu maxim) să aibă calitatea chimică, fizică şi tehnică impusă de utilizarea produsului sau de normele de calitate.

Şi în acest caz, problema (care face parte dintr-o clasă tipică — *problema amestecului sau a dietei*) se poate rezolva tot printr-un model de programare liniară.

În cazul optimizării amestecurilor de produse petroliere intervin, în afara indicatorilor economici şi a unor mărimi fizice, şi parametri specifici — cunoscuţi sub numele de constante calitative aditive — cum sînt: cifra de aciditate, conţinutul de cenuşe, conţinutul de sulf, temperatura de distilare în limite restrînse, densitatea relativă, cifra octanică, indicele Diesel, culoarea Union etc. [21].

Presupunînd că cititorul are noţiunile necesare cu privire la tehnologia petrolului, vom prezenta un exemplu numeric mai simplu (pentru a putea fi mai uşor de urmărit), preluat din [21]. Să considerăm cazul în care amestecul petrolier se realizează cu 4 benzine componente (denumite *B1*, *B2*, *B3* şi *B4*), în cantităţile şi cu caracteristicile indicate în tabela 6—3.

Scanned with OKEN Scanner

Tabela 6—3

Benzina component	Cifra octanică (M)	Densitatea la 20°C (tone/m³)	Temperatura finală de fierbere (°C)	Cantităţile disponibile (tone)	Preţul (lei/tonă)
B1	56	0,737	206	6 500	383
B2	66	0,741	204	3 600	503
B3	76	0,740	202	2 000	611
B4	90	0,749	200	8 200	701

Să zicem că se cere să se prepare 6 000 tone benzină — amestec care să satisfacă următoarele condiţii de calitate:

— cifra octanică M (determinată prin metoda Motor) să fie minimum 74;

— densitatea la 20°C să fie maximum 0,745 tone/m³;

— temperatura finală de fierbere să fie maximum 205°C.

Cu aceste date şi notînd cu x_1, x_2, x_3 şi x_4 cantităţile (în tone) din cele patru benzine componente, modelul de optimizare a amestecului pe baza criteriului minimizării preţului de cost, are următoarea formă:

$$F = 303\ x_1 + 503\ x_2 + 611\ x_3 + 701\ x_4 = minim, \qquad (6-12)$$

care este funcţia obiectiv (de minimizare a preţului de cost) scrisă cu datele indicate pe ultima coloană a tabelei 6—3;

$$\begin{cases} 56\ x_1 + 66\ x_2 + 76\ x_3 + 90\ x_4 \geqslant 74 \cdot 6\,000 & (6-13/1) \\ \frac{1}{0{,}737} x_1 + \frac{1}{0{,}741} x_2 + \frac{1}{0{,}74} x_3 + \frac{1}{0{,}749} x_4 \geqslant \frac{6\,000}{0{,}745} & (6-13/2) \\ 206\ x_1 + 204\ x_2 + 202\ x_3 + 200\ x_4 \leqslant 205 \cdot 6\,000 & (6-13/3) \\ x_1 + x_2 + x_3 + x_4 = 6\,000 & (6-13/4) \\ x_1 \leqslant 6\,500 & (6-13/5) \\ x_2 \leqslant 3\,600 & (6-13/6) \\ x_3 \leqslant 2\,000 & (6-13/7) \\ x_4 \leqslant 8\,200 & (6-13/8) \end{cases}$$

care reprezintă sistemul de restricţii. La cele de mai sus se adaugă şi condiţiile de nenegativitate:

$$x_i \geqslant 0 \qquad (i = 1,\ 2,\ 3,\ 4).$$

Restricţia (6—13/1) exprimă, cu datele din coloana a doua a tabelei 6—3, condiţia relativă la cifra octanică, care mai general (în cazul unui amestec cu n componenţi în cantităţi x_i, $i = 1, 2, \ldots, n$) se scrie în forma inegalităţii:

$$\sum_{i=1}^{n} M_i\ x_i \geqslant M_a \sum_{i=1}^{n} x_i,$$

unde M_i este cifra octanică a componenţilor, iar M_a — cifra octanică a amestecului rezultat.

Scanned with OKEN Scanner

Restricţia (6—13/2) exprimă, cu datele din coloana a treia (tabela 6—3), condiţia referitoare la densitate, care mai general se scrie în forma:

$$\sum_{i=1}^{n} \frac{x_i}{\rho_i} \geqslant \rho_a \sum_{i=1}^{n} x_i,$$

unde ρ_i sînt densităţile componenţilor, iar ρ_a — densitatea amestecului rezultat.

Restricţia (6—13/3) exprimă, cu datele din coloana a patra (tabela 6—3), condiţia referitoare la temperatura finală de fierbere, care în cazul general se scrie în forma:

$$\sum_{i=1}^{n} \theta_i \, x_i \leqslant \theta_a \sum_{i=1}^{n} x_i,$$

unde θ_i şi θ_a sînt temperaturile finale de fierbere ale componenţilor şi — — respectiv — amestecului.

Inegalităţile (6—13/4), ..., (6—13/8) exprimă restricţiile determinate de cantităţi (cerută şi disponibile).

Să observăm că modelul (6—12), (6—13) poate fi simplificat în sensul renunţării la inegalităţile $x_1 \leqslant 6500$ şi $x_4 \leqslant 8200$ deoarece amestecul final este cerut într-o cantitate $\sum_{i=1}^{4} x_i = 6000$, adică mai mică decît disponibilităţile pentru componentele x_1 şi x_4, fiecare în parte. Mai sînt posibile şi alte simplificări, însă în condiţiile rezolvării problemei prin utilizarea unui sistem de calcul automat acestea nu sînt necesare (şi, practic, nici esenţiale).

Ex. 6—4. Unul din domeniile în care programarea liniară are numeroase şi eficiente aplicaţii este acela al optimizării transporturilor, în legătură cu care se formulează clasica *problemă a transporturilor*. O vom exemplifica printr-o problemă tot din ramura petrolului (optimizarea transportului produselor petroliere). Vom considera, pentru o mai bună înţelegere, un exemplu numeric mai simplu, concret însă cu date numerice fictive şi denumiri simbolice.

Astfel, să presupunem situaţia în care cinci staţii PECO, de desfacere a produselor petroliere, denumite *S1*, *S2*, ..., *S5*, pot fi aprovizionate de la trei depozite, denumite *D1*, *D2* şi *D3*. Pentru transportul cu autocisterne a produselor petroliere de la fiecare depozit D_i ($i = 1, 2, 3$) la fiecare staţie de desfacere S_j ($j = 1, 2, \ldots, 5$) se cheltuiesc c_{ij} lei pentru un metru cub de produs; să zicem că aceste cheltuieli au valorile numerice din tabela 6—4.

Tabela 6—4

D_i Depozitul \ Staţia de desfacere S_j	*S1*	*S2*	*S3*	*S4*	*S5*
D1	3	5	2	3	2
D2	1	3	4	8	5
D3	2	8	1	2	3

Scanned with OKEN Scanner

Să mai considerăm şi următoarele valori pentru datele acestei probleme:

— cantitatea (exprimată volumetric) a produselor petroliere aflate în cele trei depozite (la un loc) este de 30 m³;

— depozitele dispun, fiecare în parte, de următoarele cantităţi de produse petroliere a_i: 16 m³ la *D1*, 8 m³ la *D2* şi 6 m³ la *D3*;

— cantităţile (volumetrice) de produse b_j cu care trebuie aprovizionate cele cinci staţii de desfacere sînt: 8 m³ pentru *S1*, 5 m³ — *S2*, 7 m³ — *S3*, 4 m³ — *S4* şi 6 m³ pentru *S5*.

Problema de optimizare care se pune într-o astfel de situaţie este aceea a determinării cantităţilor de produse x_{ij}, ce trebuie transportate de la depozitul D_i la staţia S_j, astfel încît să se asigure necesarul de produse solicitat de fiecare staţie (din disponibilităţile depozitelor), cu minimum de cheltuieli totale de transport.

Şi în această situaţie, problema se rezolvă tot printr-un model de programare liniară, care se determină (în cazul general) în următorul mod.

Există m depozite de aprovizionare şi n staţii de desfacere pentru consum. Fiecare depozit dispune de o cantitate de produse ce poate fi livrată a_i ($i = 1, 2, \ldots, m$), iar fiecare staţie de desfacere solicită o cantitate b_j ($j = 1, 2, \ldots, n$).

Dacă notăm, aşa cum am arătat mai înainte, cu x_{ij} cantitatea (necunoscută) de produs ce va fi transportată de la depozitul D_i la staţia S_j şi cu c_{ij} costul unitar (per unitatea de cantitate de produs) al transportului de la depozitul D_i la staţia S_j, atunci rezultă:

— cantitatea transportată de la un depozit D_i la cele n staţii de desfacere, care este chiar cantitatea a_i pe care o poate livra depozitul D_i, se poate exprima prin ecuaţia liniară:

$$x_{i1} + x_{i2} + \ldots + x_{in} = a_i,$$

sau pentru toate depozitele:

$$\sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1, 2, \ldots, m; \tag{6—14}$$

— cantitatea b_j achiziţionată de o staţie S_j de la cele m depozite este exprimată prin ecuaţia liniară:

$$x_{1j} + x_{2j} + \ldots + x_{mj} = b_j,$$

sau pentru toate staţiile:

$$\sum_{i=1}^{m} x_{ij} = b_j \quad j = 1, 2, \ldots, n; \tag{6—15}$$

— costul total al transportului este dat de egalitatea:

$$\begin{aligned} C_t = {} & c_{11}\, x_{11} + c_{12}\, x_{12} + \ldots + c_{1n}\, x_{1n} + \\ & + c_{21}\, x_{21} + c_{22}\, x_{22} + \ldots + c_{2n}\, x_{2n} + \\ & \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ & + c_{m1}\, x_{m1} + c_{m2}\, x_{m2} + \ldots + c_{mn}\, x_{mn} \end{aligned}$$

Scanned with OKEN Scanner

adică :

$$C_t = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}\ x_{ij} \qquad (6-16)$$

(în modelul scris în formă „desfăşurată" sumele de pe linii reprezintă costul transportului pentru produsele livrate a_i de depozitul D_i celor n staţii, iar sumele pe coloane — costul transportului pentru achiziţionarea cantităţii b_j de către o staţie S_j de la cele m depozite furnizoare).

Este evident că valorile cantităţilor de produse transportate nu pot fi negative ; deci trebuie respectate şi în acest caz condiţiile de nenegativitate :

$$x_{ij} \geqslant 0 \quad i = 1, 2, \ldots, m, \ j = 1, 2, \ldots, n. \qquad (6-17)$$

În această situaţie, determinarea necunoscutelor x_{ij} din sistemul de ecuaţii liniare (6—14) şi (6—15) conduce la o infinitate de soluţii. Soluţia considerată ca optimă va fi aceea care asigură cele mai mici cheltuieli totale de transport ($C_t = min$). De aici rezultă următoarea formă de program liniar :

$$C_t = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}\ x_{ij} = min, \qquad (6-18)$$

care reprezintă funcţia obiectiv,

$$\begin{cases} \sum_{j=1}^{n} x_{ij} = a_i \ \ i = 1, 2, \ldots, m & (6-19/1) \\ \sum_{i=1}^{m'} x_{ij} = b_j \ \ j = 1, 2, \ldots, n & (6-19/2) \\ x_{ij} \geqslant 0 \quad i = 1, 2, \ldots, m, \ j = 1, 2, \ldots, n & (6-19/3) \end{cases}$$

care reprezintă sistemul de restricţii.

Modelul (6—18), (6—19) este cunoscut sub numele de *program de transport*.

În cazul particular al exemplului cu datele din tabela 6—4, pentru care $m = 3$ şi $n = 5$, programul de transport are următoarea formă :

$$\begin{aligned} C_t = {} & 3x_{11} + 5x_{12} + 2x_{13} + 3x_{14} + 2x_{15} + \\ & + 1x_{21} + 3x_{22} + 4x_{23} + 8x_{24} + 5x_{25} + \\ & + 2x_{31} + 8x_{32} + 1x_{33} + 2x_{34} + 3x_{35} = min \end{aligned} \qquad (6-20)$$

Scanned with OKEN Scanner

(funcţia obiectiv) şi

$$\begin{cases} x_{11} + x_{12} + x_{13} + x_{14} + x_{15} = 16 \\ x_{21} + x_{22} + x_{23} + x_{24} + x_{25} = 8 \\ x_{31} + x_{32} + x_{33} + x_{34} + x_{35} = 6 \\ x_{11} + x_{21} + x_{31} = 8 \\ x_{12} + x_{22} + x_{32} = 5 \\ x_{13} + x_{23} + x_{33} = 7 \\ x_{14} + x_{24} + x_{34} = 4 \\ x_{15} + x_{25} + x_{35} = 6 \\ x_{11} \geqslant 0 \\ x_{12} \geqslant 0 \\ x_{13} \geqslant 0 \\ x_{14} \geqslant 0 \\ x_{15} \geqslant 0 \\ x_{21} \geqslant 0 \\ x_{22} \geqslant 0 \\ x_{23} \geqslant 0 \\ x_{24} \geqslant 0 \\ x_{25} \geqslant 0 \\ x_{31} \geqslant 0 \\ x_{32} \geqslant 0 \\ x_{33} \geqslant 0 \\ x_{34} \geqslant 0 \\ x_{35} \geqslant 0 \end{cases} \qquad (6-21)$$

(sistemul de restricţii de $m + n = 8$ egalităţi şi $m \cdot n = 15$ inegalităţi, cu $m \cdot n = 15$ necunoscute x_{ij}).

Ex. 6—5. Vom încheia şirul acestor exemple cu un caz care va ilustra marea diversitate a posibilităţilor de utilizare practică a programării liniare.

După cum se ştie, legătura dintre sapa de foraj cu trei conuri şi garnitura de antrenare (coloana de prăjini) se face printr-un filet special, cu o tehnologie de execuţie complexă, în care un loc însemnat îl ocupă şi prelucrările prin aşchiere. Optimizarea acestei operaţii de filetare la sapa de foraj cu trei conuri, care are un efect sensibil atît asupra procesului tehnologic de fabricare a sapei cît şi asupra performanţelor ce se vor obţine la forajul sondelor, se poate realiza pe baza unor criterii economice şi tehnice folosind parametri ca:

— adîncimea de aşchiere (t);
— avansul la filetare (s);
— viteza de rotaţie a piesei (n) în timpul filetării cepului sapei;
— viteza de aşchiere;
— geometria, dimensiunile piesei filetate şi materialul piesei;
— durabilitatea cuţitelor;
— costul operaţiilor de aşchiere;
— „ritmul" liniei tehnologice;
— cinematica şi dinamica maşinii unelte;
— rezistenţa şi rigiditatea elementelor sistemului tehnologic elastic;
— calitatea suprafeţei prelucrate etc.

Scanned with OKEN Scanner

Modelarea regimului de aşchiere la o maşină unealtă pentru executat filetele la sapele cu trei conuri (din cadrul unui lanţ tehnologic fără incidente de gîtuire) se poate face cu relaţiile de mai jos, care se obţin prin analiza procesului tehnologic [6]:

$$C_t = \left(\frac{1\,000\ c_v}{\pi d}\right)^{\frac{\beta}{m}} \cdot \frac{c_1 \cdot l \cdot z \cdot \alpha}{n^{\frac{m+\beta}{m}} \cdot t^{\frac{m+X_v \cdot \beta}{m}} \cdot s^{\frac{m+Y_v \cdot \beta}{m}}}, \qquad (6-22)$$

care exprimă costul operaţiilor de filetare;

$$n \cdot t \cdot s \geqslant \frac{l \cdot z}{R_l \cdot Q_{min} \cdot K_i - \tau_1}, \qquad (6-23)$$

relaţie care modelează „ritmul" liniei de fabricaţie;

$$n \cdot t^{X_{F_Z}} \cdot s^{Y_{F_Z}} \leqslant \frac{1\,000\ N}{\pi d\, c_N}, \qquad (6-24)$$

inegalitate care rezultă din calculul puterii necesare pentru filetare;

$$t^{X_{F_Z}} \cdot s^{Y_{F_Z}} \leqslant \frac{\sigma_a \cdot \beta^2}{10\, \xi c_{F_z}}, \qquad (6-25)$$

care ţine seama de rezistenţa şi stabilitatea sculei;

$$t^{X_{F_y}} \cdot s^{Y_{F_y}} \leqslant \frac{F_a}{c_{F_y}}, \qquad (6-26)$$

condiţie care rezultă din încărcarea maximă admisă de mecanismul de avans;

$$s^{1,07} \leqslant \frac{R_z \cdot r^{0,65}}{0,21}, \qquad (6-27)$$

condiţie referitoare la calitatea suprafeţei prelucrate;

$$t_0 \leqslant t \leqslant z, \qquad (6-28)$$

inegalităţi ce rezultă din condiţiile referitoare la adausul de prelucrare şi la adîncimea minimă de aşchiere;

$$\begin{aligned} n_{min} &\leqslant n \leqslant n_{max}, \\ s_{min} &\leqslant s \leqslant s_{max}, \end{aligned} \qquad (6-29)$$

care exprimă condiţiile (limitările) impuse de cinematica maşinii de filetare.

Semnificaţiile notaţiilor din relaţiile (6—22), ..., (6—29), ca şi valorile concrete (referitoare la un caz studiat la Întreprinderea „1 Mai" Ploieşti), sînt următoarele:

Scanned with OKEN Scanner

C_t — costul total (în lei) al executării filetului, considerat ca criteriu de optimizare (se va determina situaţia în care $C_t = min$);

c_1 — retribuţia muncitorului care lucrează la maşina de filetat şi cota parte din regia secţiei ($c_1 = 0{,}576$ lei/min);

d — diametrul piesei supuse prelucrării ($d = 114{,}3$ mm);

l — lungimea teoretică de prelucrare a piesei ($l = 128{,}16$ mm);

z — adausul de prelucrare, îndepărtat de pe piesă prin aşchiere ($z = 0{,}324$ mm);

n — viteza de rotaţie a piesei în timpul prelucrării ei (în rot/min), necunoscută ce trebuie determinată în condiţiile minimizării costului prelucrării;

t — adîncimea de aşchiere (în mm), necunoscută ce trebuie determinată în condiţiile minimizării costului prelucrării;

s — avansul la filetare (în mm/rot), necunoscută ce trebuie determinată în condiţiile minimizării costului prelucrării;

m — exponentul durabilităţii sculei (cuţitului utilizat la prelucrarea piesei), durabilitatea fiind definită ca intervalul de timp între două ascuţiri ale sculei pentru aşchiere. Considerînd un cuţit prevăzut cu plăcuţă din aliaj dur (P10) şi piese supuse prelucrării confecţionate din oţel crom-nichel, rezultă $m = 0{,}2$ [6];

α şi β — parametri care intervin în modelele pentru analiza durabilităţii sculei de prelucrare. În condiţiile de material, tip de cuţit şi elemente geometrice specificate mai sus, din [6] rezultă: $\alpha = 1{,}613$ şi $\beta = 0{,}317$;

c_v — coeficient tabelat (pentru tipul de cuţit şi de material utilizat rezultă $c_v = 237$);

X_v, Y_v — coeficienţi tabelaţi care intervin în expresia vitezei de aşchiere (pentru tipul de cuţit şi de material considerate rezultă: $X_v = 0{,}15$ şi $Y_v = 0{,}2$);

R_l — „ritmul" liniei tehnologice, care se calculează prin raportul dintre fondul real anual de timp şi planul anual de producţie al liniei ($R_l = 13{,}3$ min/sapă);

Q_{min} — numărul minim de maşini în funcţiune simultană din linia tehnologică ($Q_{min} = 2$);

K_i — coeficient real de încărcare al maşinilor de prelucrare ($K_i = 0{,}85$);

τ_1 — timpul auxiliar, nelucrător, al sculei pentru o piesă prelucrată ($\tau_1 = 2{,}5$ min/sapă);

N — puterea necesară prelucrării ($N = 5{,}8$ kW);

c_N — coeficient tabelat care corectează valoarea puterii de bază necesară pentru prelucrare, în funcţie de material, tip de cuţit, unghi de atac la aşchiere etc. (în cazul considerat rezultă $c_N = 0{,}061$);

X_{F_z} şi Y_{F_z} — coeficienţi tabelaţi care intervin în relaţiile pentru calculul componentei F_z a forţei de aşchiere la filetare (în funcţie de material, de tipul cuţitului etc. rezultă: $X_{F_z} = 1$ şi $Y_{F_z} = 0{,}75$);

σ_a — rezistenţa admisibilă la încovoiere a materialului sculei ($\sigma_a = 15 \cdot 10^7$ N/m²);

ξ — coeficient care se calculează în funcţie de secţiunea de fixare a sculei ($\xi = 47{,}4$);

Scanned with OKEN Scanner

c_{F_z} – coeficient de corecție din expresia de calcul a componentei F_z a forței de așchiere ($c_{F_z} = 300$);

X_{F_y} și Y_{F_y} – coeficienți tabelați care intervin în relația pentru calculul componentei F_y a forței de așchiere la filetare (în cazul considerat rezultă $X_{F_y} = 0{,}9$ și $Y_{F_y} = 0{,}6$);

F_a – forța maximă admisă de mecanismul de avans ($F_a = 188$ N);

c_{F_y} – coeficient de corecție din relația de calcul a componentei F_y a forței de așchiere ($c_{F_y} = 243$);

R_z – rugozitatea suprafeței piesei prelucrare (înălțimea microneregularităților). Considerăm $R_z = 1{,}6$ μm;

r – raza de rotunjire a vîrfului cuțitului (se consideră $r^{0,65} = 0{,}508$ mm);

t_0 – adîncimea minimă de așchiere ($t_0 = 0{,}162$ mm);

$n_{min} = 40$ rot/min și $n_{max} = 1400$ rot/min sînt limitele (minimă și maximă) ale vitezei de rotație a piesei în timpul prelucrării;

$s_{min} = 0{,}05$ mm/rot și $s_{max} = 0{,}9$ mm/rot – limitele (minimă și maximă) ale avansului la filetare.

Cu aceste valori, modelul (6–22), (6–23), ..., (6–29), de optimizare a executării filetului la sapele cu trei role, ia forma:

$$C_t = 660^{-1,585} \cdot \frac{38{,}1}{n^{-0,585} \cdot t^{0,762} \cdot s^{0,683}} = min, \qquad (6-30)$$

$$\begin{cases} n.t.s \geqslant 2{,}064 \\ n.t.s^{0,75} \leqslant 265 \\ t.s^{0,75} \leqslant 106 \\ t^{0,9} \cdot s^{0,5} \leqslant 0{,}773 \\ s^{1,07} \leqslant 3{,}87 \\ 0{,}162 \leqslant t \leqslant 0{,}324 \\ 40 \leqslant n \leqslant 1400 \\ 0{,}05 \leqslant s \leqslant 0{,}9 \end{cases} \qquad (6-31)$$

După cum se vede, modelul (6–30), (6–31) nu este liniar. El poate fi însă „liniarizat" prin logaritmare. Dacă logaritmăm și facem schimbările de variabile:

$$\begin{aligned} C_t &= \lg C_t, \\ x_1 &= \lg n, \\ x_2 &= \lg (100\ t) \text{ sau } \lg t = x_2 - 2 \end{aligned} \qquad (6-32)$$

și

$$x_3 = \lg(100\ s) \text{ sau } \lg s = x_3 - 2,$$

atunci modelul (6–30), (6–31) se transformă în următorul program liniar:

$$C_t = 0{,}585\ x_1 - 0{,}762\ x_2 - 0{,}683\ x_3 - 0{,}098 = min, \qquad (6-33)$$

Scanned with OKEN Scanner

care reprezintă funcţia obiectiv;

$$\begin{cases} x_1 + x_2 + x_3 \geqslant 4{,}314 \\ x_1 + x_2 + 0{,}75\, x_3 \leqslant 5{,}923 \\ x_2 + 0{,}75\, x_3 \leqslant 5{,}525 \\ 0{,}9\, x_2 + 0{,}6\, x_3 \leqslant 2{,}888 \\ 1{,}07\, x_3 \leqslant 2{,}728 \\ x_1 \leqslant 3{,}146 \\ x_1 \geqslant 1{,}602 \\ x_2 \leqslant 1{,}511 \\ x_2 \geqslant 1{,}207 \\ x_3 \leqslant 1{,}954 \\ x_3 \geqslant 0{,}699 \end{cases} \qquad (6-34)$$

sistem cu 11 inegalităţi-egalităţi ce exprimă restricţiile, la care mai trebuie adăugate şi condiţiile de strictă nenegativitate $x_1 > 0$, $x_2 > 0$ şi $x_3 > 0$.*

* * *

Din exemplele de mai sus rezultă că programarea liniară este un procedeu matematic de optimizare a proceselor (sistemelor) ce pot fi descrise prin expresii liniare. Ea constă, în esenţă, în determinarea maximului/minimului unei funcţii (funcţia obiectiv) de mai multe variabile necunoscute, legate între ele printr-un sistem cu un număr oarecare de ecuaţii şi inecuaţii (care reprezintă restricţiile problemei). Atît funcţia obiectivă cît şi sistemul de restricţii sînt forme liniare şi alcătuiesc modelul de programare liniară al procesului analizat.

6.2. METODE DE REZOLVARE A PROGRAMELOR LINIARE

După formularea modelului de programare liniară urmează o etapă esenţială a studiului de optimizare — aceea a rezolvării problemei de programare liniară, adică a determinării mulţimii de valori numerice x_i ($i = 1, 2, \ldots, n$) care satisfac restricţiile şi determină o valoare maximă/minimă a funcţiei obiectiv.

În continuare vom face o succintă prezentare a principalelor metode de rezolvare a programelor liniare, în forma algoritmică specifică utilizării sistemelor de calcul automat.

Generalităţi

Pentru evitarea confuziilor, vom preciza — mai întîi — înţelesul pe care îl are noţiunea de „soluţie" în cazul problemelor de programare liniară:

— prin *soluţie* a problemei se înţelege mulţimea de $n + m$ valori x_j care satisfac sistemul de ecuaţii (6—2 *a*), al restricţiilor;

— dacă, în plus, această soluţie verifică şi condiţiile de nenegativitate (6—3*a*), adică dacă $x_j \geqslant 0$ ($j = 1, 2, \ldots, n + m$), atunci soluţia se

* Acest exemplu ne-a fost propus de dr. ing. V. Ulmanu, de la Catedra de utilaj petrolier din Institutul de petrol şi gaze.

Scanned with OKEN Scanner

numeşte *soluţie posibilă* (sau soluţie realizabilă sau, încă, soluţie admisibilă);

— dacă restricţiile sînt liniar independente iar soluţia modelului conţine m componente x_i nenule ($x_i \geqslant 0$, $i = 1, 2, \ldots, m$) şi n componente nule ($x_j = 0$, $j = 1, 2, \ldots, n$), atunci soluţia poartă numele de soluţie posibilă de bază sau — mai scurt — de *soluţie de bază* (în cazurile în care sînt mai puţin decît m valori x_i nenule, soluţia se numeşte *soluţie de bază degenerată*);

— soluţia de bază care face ca funcţia obiectiv (6—1*a*) să fie maximă/minimă este denumită soluţie de bază optimală sau, direct, *soluţie optimală*.

Să considerăm, pentru a ne reaminti cîteva noţiuni fundamentale, că modelul are N variabile şi M restricţii; atunci toate valorile soluţiei care fac ca funcţia obiectiv să fie $F = max$ (sau $F = min$) vor corespunde vîrfurilor poliedrului convex format din intersecţia N-edrului de referinţă (care exprimă condiţiile de nenegativitate $x_i \geqslant 0$) cu hiperplanele reprezentînd restricţiile.* Deci, pentru a se obţine un maximum/minimum al funcţiei obiectiv, orice soluţie trebuie să îndeplinească următoarea condiţie necesară: „să conţină cel puţin $N-M$ variabile nule". Notînd: $N = n + m$ şi $M = m$, rezultă că această condiţie are şi enunţul: „să conţină cel puţin n variabile nule".

Este esenţial să reţinem că toate poliedrele astfel construite sînt convexe (poliedrul format prin intersecţia hiperplanelor reprezentînd restricţiile cu N-edrul de referinţă este convex deoarece intersecţia a două poliedre convexe este tot un poliedru convex). În acest fel, „deplasarea" din vîrf în vîrf şi, corespunzător, creşterile funcţiei F (sau micşorările ei) vor conduce sigur la atingerea maximului/minimului funcţiei obiectiv. Dacă domeniul nu ar fi convex, atunci nu ar fi certă obţinerea unui F maxim maximorum/minim minimorum.

Ex. 6—6. Pentru a ilustra cele de mai sus vom reveni asupra programului „mai simplu" (cu numai două variabile) din *Ex. 6—1*, (6—4), (6—5), şi anume:

$$\begin{cases} x_1 \geqslant 0 \\ x_2 \geqslant 0 \\ 3x_1 + 5x_2 \leqslant 15 \\ 6x_1 + 2x_2 \leqslant 24 \end{cases} \qquad (6-5)$$

$$F = 200\ x_1 + 100\ x_2 = max. \qquad (6-4)$$

Modelul fiind bidimensional, restricţiile reprezintă semiplane care într-un sistem de axe de coordonate rectangular x_1Ox_2 pot fi redate aşa ca în figura 6.1, unde:

— condiţiile de nenegativitate ($x_1 \geqslant 0$ şi $x_2 \geqslant 0$) se reprezintă prin semiplanele care conţin sensul pozitiv al axelor şi sînt limitate de axele

* Această afirmaţie este riguroasă dacă maximul/minimul funcţiei obiectiv F este atins într-un singur punct. Evident, sînt posibile cazuri degenerescente în care maximul/minimul este atins în toate punctele unei feţe sau unei linii de intersecţie a poliedrului.

Scanned with OKEN Scanner

de coordonate $x_1 = 0$ şi — respectiv — $x_2 = 0$ (ca urmare, semiplanele $x_1 > 0$ şi $x_2 < 0$ sînt haşurate în figura 6.1);

— semiplanul $3x_1 + 5x_2 \leqslant 15$ este partea nehaşurată delimitată de dreapta (AB) care are ecuaţia $3x_1 + 5x_2 = 15$;

— restricţia $6x_1 + 2x_2 \leqslant 24$ reprezintă semiplanul nehaşurat delimitat de dreapta (BC), adică de $6x_1 + 2x_2 = 24$.

Dreptele care delimitează semiplanele restricţiilor, adică (OA) $x_1 = 0$, (AB) $3x_1 + 5x_2 = 15$, (BC) $6x_1 + 2x_2 = 24$ şi (OC) $x_2 = 0$, formează poligonul convex $OABC$. Punctele situate în interiorul sau pe laturile acestui poligon au coordonatele (x_1, x_2) ce reprezintă soluţii ale sistemului de restricţii (6—5).

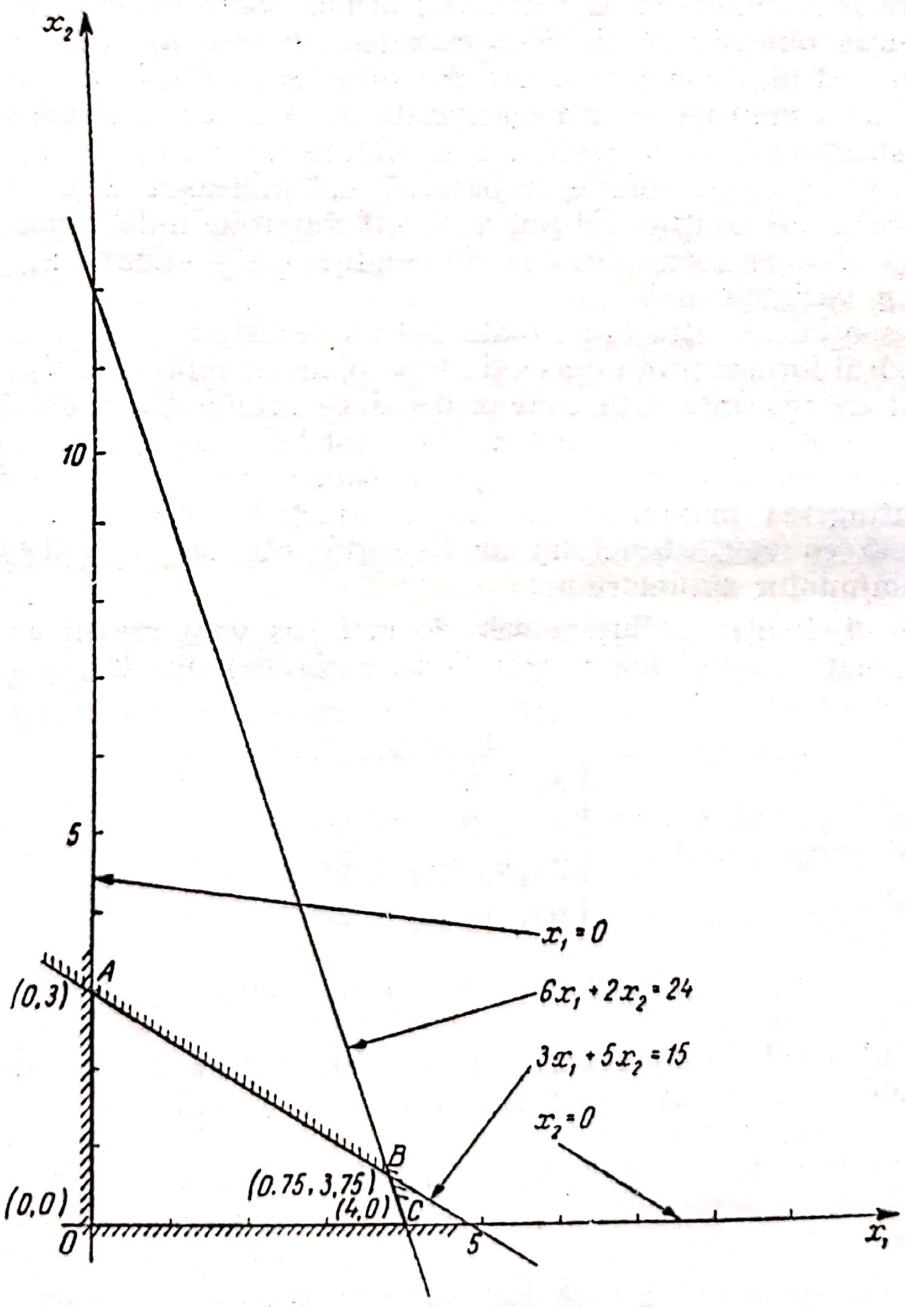


Fig. 6.1

Scanned with OKEN Scanner

Într-adevăr, dacă introducem variabilele de ecart x_{n+i}, $i = 1,2$ (unde $n = 2$) obținem următorul sistem de ecuații:

$$\begin{cases} 3x_1 + 5x_2 + x_3 = 15 \\ 6x_1 + 2x_2 + x_4 = 24 \end{cases} \qquad (6-5/1)$$

care are în vedere și condițiile de nenegativitate $x_i \geqslant 0$ ($i = 1, 2, 3, 4$). Sistemul (6—5/1) este format din $m = 2$ ecuații, cu $n = 4$ variabile. După cum se știe, un astfel de sistem are o infinitate de soluții.

Pentru a obține o soluție a sistemului (6—5/1) putem alege valori pentru două variabile, rezultînd astfel un sistem de 2 ecuații cu 2 necunoscute care (dacă este posibil!) se rezolvă în mod obișnuit.

Un procedeu simplu de a alege valori inițiale pentru cîte două necunoscute constă în a le anula; în acest fel putem obține cel mult $C_4^2 = 6$ soluții, și anume:

Soluția	x_1	x_2	x_3	x_4	Comentarii
1	0	0	15	24	Soluție de bază (are două componente nenule)
2	0	3	0	18	Soluție de bază (are două componente nenule)
3	0	12	−45	0	Nu este posibilă (deoarece $x_3 < 0$)
4	5	0	0	−6	Nu este posibilă (deoarece $x_4 < 0$)
5	4	0	13	0	Soluție de bază (are două componente nenule)
6	3,75	0,75	0	0	Soluție optimală (face ca $F = max.$)

Dintre aceste soluții, constatăm că nu sînt posibile soluțiile 3 și 4 (deoarece în cazul lor nu sînt respectate condițiile de nenegativitate), restul soluțiilor (adică 1, 2, 5 și 6) fiind posibile (pentru toate aceste soluții $x_i \geqslant 0$).

Verificăm, prin acest exemplu, cele afirmate mai înainte și anume că soluțiile posibile, obținute prin acest procedeu, au proprietatea că reprezintă vîrfurile (x_1, x_2) ale poligonului convex $OABC$ (fig. 6.1): $O = (0,0)$ — soluția *1*, $A = (0,3)$ — soluția *2*, $B = (3,75, 0,75)$ — soluția *6* și $C = (4,0)$ — soluția *5*. Aceste soluții avînd și cîte $m = 2$ componente nenule reprezintă soluții posibile de bază.

În sfîrșit, dintre soluțiile de bază, una — și anume soluția *6* — este soluția optimală, deoarece face ca funcția obiectiv

$$F = 200x_1 + 100x_2 + 0 \cdot x_3 + 0 \cdot x_4 \qquad (6-4/1)$$

să aibă valoarea maximă:

x_1	x_2	x_3	x_4	F
0	0	15	24	0
0	3	0	18	300
4	0	13	0	800
3,75	0,75	0	0	825 → *max*

* * *

Scanned with OKEN Scanner

Încheiem acest paragraf prin a repeta o observaţie importantă, prezentată anterior, în legătură cu interpretarea soluţiei unui program liniar ca punct în spaţiul euclidian N-dimensional:

— mulţimea planelor corespunzătoare unui sistem de restricţii de forma (6—2a) este fie mulţimea vidă, fie un poliedru n-dimensional;

— vîrfurile poliedrului corespund planelor de referinţă $x_i \geqslant 0$ şi reciproc;

— dacă problema are soluţie mărginită, atunci există un vîrf al poliedrului pentru care valoarea funcţiei obiectiv este maximă/minimă.

Deci acest vîrf este soluţia optimală a problemei (mai mult, în cazul existenţei mai multor soluţii, toate sînt combinaţiile convexe ale unui număr de vîrfuri-soluţie). Ca urmare, este suficient să căutăm soluţia problemei printre vîrfurile poliedrului.

De aici, şi ţinînd seama de faptul că poliedrul are un număr finit de vîrfuri (C_n^m), rezultă că un număr finit de încercări va duce la găsirea soluţiei optimale. Astfel înţelegem de ce scopul diferitelor metode de rezolvare a problemelor de programare liniară este acelaşi: reducerea numărului de încercări menit să conducă la soluţia optimală. Pentru aceasta, în cadrul unei încercări, se ia în considerare numai un plan care „îmbunătăţeşte" valoarea funcţiei obiectiv, faţă de valoarea ei pentru planul considerat în încercarea precedentă etc. Prin urmare, metodele de rezolvare a problemelor de programare liniară sînt metode iterative care — în esenţă — constau în căutarea unui plan de plecare (iniţial) şi în găsirea unui alt plan pentru care valoarea funcţiei obiectiv se îmbunătăţeşte (în sensul definit de program: maximizare sau minimizare). De obicei, noul plan este vecin cu precedentul, adică vîrfuri vecine pe poliedrul soluţiilor. Diferenţa esenţială între diversele metode constă în criteriul de alegere a noului plan.

Metoda simplex

Principiul metodei constă în a calcula, plecînd de la o soluţie de bază, o altă soluţie de bază care îmbunătăţeşte valoarea dată de funcţia obiectiv (în sensul maximizării sau minimizării ei), optimul fiind „atins" atunci cînd valoarea acestei funcţii nu mai poate fi îmbunătăţită în continuare.*

Procedeul este, deci, iterativ, principalele etape de calcul fiind următoarele:

1° determinarea iniţială a unei soluţii de bază („baza iniţială");

2° trecerea la altă soluţie de bază, plecînd de la baza iniţială/precedentă (adică trecerea de la un vîrf la altul al poliedrului soluţiilor), astfel încît să se obţină o valoare mai mare/mică a funcţiei obiectiv;

3° considerarea ca soluţie optimală a soluţiei de bază obţinute după etapa 2°, atunci cînd nu mai este posibil să se mărească/micşoreze valoarea funcţiei obiectiv.

---

* Algoritmul simplex a fost elaborat în 1947 de G. B. Dantzig, pe baza unei observaţii a matematicianului Von Neumann.

Scanned with OKEN Scanner

Pentru a arăta modul general de aplicare a acestui algoritm, să considerăm modelul matricial al unei probleme de programare liniară exprimat în forma standard (cu variabile de ecart):

$$F = \mathbf{CX} = max,$$
$$\mathbf{AX} = \mathbf{B},$$
$$\mathbf{X} \geqslant 0,$$

unde, față de expresiile (6—1*b*), (6—2*b*) și (6—3*b*), matricele **C**, **A** și **X** includ, prin „alipire", și elementele referitoare la variabilele de ecart, adică:

$$\mathbf{C} = (c_1 c_2 \ \ldots \ c_n c_{n+1} \ldots \ c_{n+m}),$$

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \\ x_{n+1} \\ \vdots \\ x_{n+m} \end{bmatrix},$$

$\mathbf{A} = (a_{ij})$ în care porțiunea $i = 1, 2, \ldots, m$ și $j = n+1, n+2, \ldots, n+m$ reprezintă matricea unitate.

Să presupunem că avem o soluție de bază formată din m variabile nenegative și nu toate nule, x_i, numerotate de la $n+1$ la $n+m$. Se poate scrie:

$$\sum_{i=n+1}^{n+m} \mathbf{A}_i x_i = \mathbf{B} \qquad (6—35)$$

și

$$F = \sum_{i=n+1}^{n+m} c_i x_i, \qquad (6—36)$$

cu $x_i \geqslant 0$, $i = n+1, n+2, \ldots, n+m$, în care

$$\mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix} \quad \text{și} \quad \mathbf{A}_i = \begin{bmatrix} a_{1i} \\ a_{2i} \\ \vdots \\ a_{mi} \end{bmatrix}.$$

Deci, matricele $\mathbf{A}_i$ sînt m vectori coloană ai matricei dreptunghiulare **A**. Celelalte n coloane $\mathbf{A}_j$, $j = 1, 2, \ldots, n$, pot fi exprimate prin combinații liniare în funcție de cei m vectori coloană $\mathbf{A}_i$, care formează o bază a spațiului vectorial m-dimensional, adică prin relații de forma:

$$\mathbf{A}_j = \sum_{i=n+1}^{n+m} x_{ij} \mathbf{A}_i, \quad j = 1, 2, \ldots, n. \qquad (6—37)$$

Scanned with OKEN Scanner

Coeficienţii combinaţiilor liniare (6—37) pot fi calculaţi prin inversarea matricelor pătrate formate din cei m vectori (coloane) de bază $\mathbf{A}_i$, determinaţi cu (6—35). Vom putea scrie, de asemenea:

$$F_j = \sum_{i=n+1}^{n+m} x_{ij} c_i, \quad j = 1, 2, \ldots, n. \tag{6—38}$$

În continuare, să notăm cu θ o cantitate scalară pozitivă. Alegînd unul din vectorii $\mathbf{A}_j$, care nu sînt în bază, şi multiplicînd egalitatea (6—37) cu θ vom putea scrie:

$$\theta \mathbf{A}_j = \sum_{i=n+1}^{n+m} \theta x_{ij} \mathbf{A}_i,$$

cu care obţinem o nouă expresie pentru (6—35) şi anume:

$$\sum_{i=n+1}^{n+m} \mathbf{A}_i x_i = \mathbf{B} = \sum_{i=n+1}^{n+m} (x_i - \theta x_{ij}) \mathbf{A}_i + \theta \mathbf{A}_j. \tag{6—39}$$

De asemenea, adăugăm celor doi membri ai relaţiei (6—36) termenul $\theta(c_j - F_j)$ şi obţinem:

$$F + \theta(c_j - F_j) = \sum_{i=n+1}^{n+m} c_i x_i + \theta(c_j - F_j). \tag{6—40}$$

Deoarece din (6—38), prin înmulţire cu θ, rezultă:

$$\theta F_j = \sum_{i=n+1}^{n+m} \theta x_{ij} c_i,$$

atunci (6—40) se poate scrie în forma:

$$F + \theta(c_j - F_j) = \sum_{i=n+1}^{n+m} (x_i - \theta x_{ij}) c_i + \theta c_j. \tag{6—41}$$

Dacă cel puţin unul din coeficienţii x_{ij} este strict pozitiv, atunci putem alege un $\theta > 0$ astfel ca:

$$\theta = \theta_0 = \min_i \frac{x_i}{x_{ij}}.$$

Toţi termenii $(x_i - \theta x_{ij})$ vor fi strict pozitivi, în afară de unul care va fi nul şi va corespunde lui θ_0 definit mai sus. Ca urmare, vom extrage din bază vectorul $\mathbf{A}_i$ astfel ca:

$$\theta = \theta_0 = \min_i \frac{x_i}{x_{ij}},$$

pentru că coeficientul său se va anula în ecuaţia (6—39); el va fi înlocuit prin $\mathbf{A}_j$, cu indicele j din expresia lui θ_0, adică cu cel j pentru care $(c_j - F_j) = max$ şi pozitiv. Noua soluţie de bază astfel obţinută este, deci, distinctă de precedenta; astfel am trecut la un alt punct extrem, adică la un alt vîrf al poliedrului soluţiilor de bază.

Scanned with OKEN Scanner

Această nouă soluție va trebui să corespundă unei creșteri a valorii funcției obiectiv (în cazul maximizării, presupusă inițial), ceea ce va avea loc dacă vom alege vectorul cu indicele j — care intră în bază — astfel ca:

$$(c_j - F_j) > 0.$$

Dacă problema este de minimizare a lui F, vom considera:

$$(F_j - c_j) > 0.$$

Alegînd cantitatea $(c_j - F_j) > 0$ cea mai mare posibilă, vom selecționa vectorul $\mathbf{A}_j$ (cu același indice j) și vom putea determina o transformare a soluției precedente astfel încît să obținem cea mai mare creștere posibilă a valorii funcției obiectiv.

Dacă, după o serie de calcule succesive, nu se va mai putea găsi o cantitate $(c_j - F_j) > 0$, atunci funcția obiectiv F nu va putea să mai crească, ceea ce înseamnă că maximul acestei funcții a fost atins. În cazul $F = min$, minimul a fost atins atunci cînd nu se va mai putea găsi o cantitate $(F_j - c_j) > 0$.

Să observăm că aceste cantități $c_j - F_j$ reprezintă variațiile unitare ale funcției obiectiv, la trecerea de la un punct extrem (vîrf al poliedrului) la altul; sînt, de fapt creșterile variabilelor x_j „în afara" bazei, în raport cu soluția de bază.

Rezultă deci că alegînd un $\mathbf{A}_j$ astfel încît $(c_j - F_j) > 0$ să fie cea mai mare posibilă și selectînd linia i pentru care $\theta = x_i/x_{ij}$ să fie minimă (însă pozitivă sau nulă), se va putea determina o transformare a soluției inițiale astfel încît să se obțină cea mai mare creștere posibilă (eventual nulă) a funcției obiectiv F. Maximul funcției F va fi atins atunci cînd nu mai este posibil să obținem o creștere a lui F, adică atunci cînd nu mai poate fi găsită nici o cantitate $c_j - F_j$ pozitivă.

Ex. 6—7. Pentru a ilustra cele de mai sus, vom considera un exemplu numeric mai simplu, care să poată fi prezentat în întregime în cadrul spațiului limitat de care dispunem. Se dă, deci, următoarea problemă de programare liniară:

să se găsească maximul funcției

$$F = 60\ x_1 + 60\ x_2 + 90\ x_3 + 90\ x_4$$

în cazul restricțiilor:

$$\begin{cases} x_1 + x_2 + x_3 + x_4 \leqslant 15 \\ 7x_1 + 5x_2 + 3x_3 + 2x_4 \leqslant 1 \\ 3x_1 + 5x_2 + 10x_3 + 15x_4 \leqslant 1 \end{cases}$$

cu respectarea condițiilor de nenegativitate $x_i \geqslant 0$ $(i = 1, 2, 3, 4)$.

Introducînd variabilele de ecart x_5, x_6 și x_7, modelul acestei probleme este:

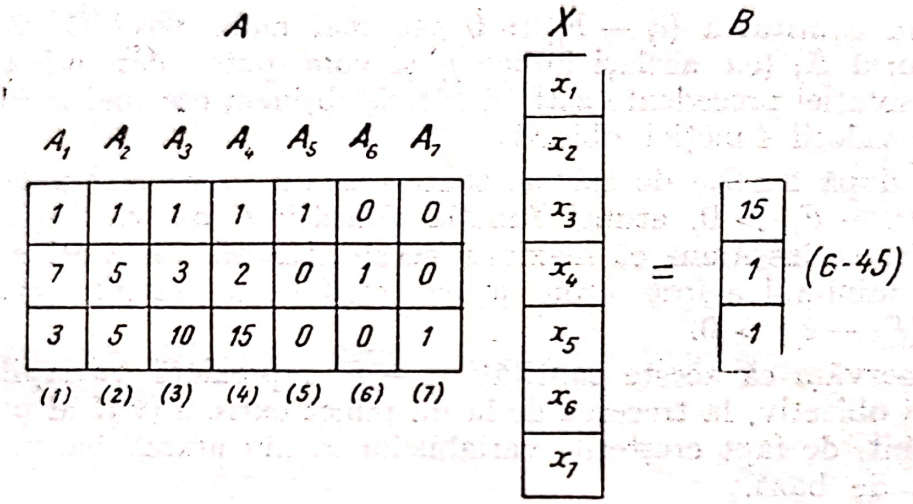
$$F = 60\ x_1 + 60\ x_2 + 90\ x_3 + 90\ x_4 + 0 \cdot x_5 + 0 \cdot x_6 + 0 \cdot x_7 = max, \tag{6—42}$$

Scanned with OKEN Scanner

$$\begin{cases} x_1 + [x_2 + x_3 + x_4 + x_5 = 15 \\ 7x_1 + 5x_2 + 3x_3 + 2x_4 + x_6 = 1 \\ 3x_1 + 5x_2 + 10x_3 + 15x_4 + x_7 = 1 \end{cases} \quad (6-43)$$

$$x_i \geqslant 0 \ (i = 1, 2, \ldots, 7). \quad (6-44)$$

Sistemul de ecuaţii (6—43), reprezentînd restricţiile, poate fi scris vectorial în forma:



O soluţie de bază evidentă este:

$$x_1^{(1)} = x_2^{(1)} = x_3^{(1)} = x_4^{(1)} = 0, \ x_5^{(1)} = 15, \ x_6^{(1)} = 1 \text{ şi } x_7^{(1)} = 1,$$

pentru care funcţia obiectiv (6—42) ia valoarea $F^{(1)} = 0$. Notăm această primă soluţie, din prima iteraţie, cu (1) plasat ca exponent.

Alegînd soluţia iniţială de mai sus, înseamnă că am format baza din coloanele (5), (6) şi (7); ca urmare, plecînd de la (6—45), expresiile (6—37 se pot scrie, în cazul acestui exemplu, sub forma:

$$\begin{matrix} & \mathbf{A}_j & & & & \mathbf{A}_i & & \\ \end{matrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 7 & 5 & 3 & 2 \\ 3 & 5 & 10 & 15 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_{51} & x_{52} & x_{53} & x_{54} \\ x_{61} & x_{62} & x_{63} & x_{64} \\ x_{71} & x_{72} & x_{73} & x_{74} \end{bmatrix},$$

(1) (2) (3) (4) (5) (6) (7)

pentru care $n = 4$, $m = 3$ şi, evident, $n + m = 7$. De aici rezultă imediat valorile coeficienţilor x_{ij} ($i = 1, 2, 3$ şi $j = 1, 2, 3, 4$):

$$\begin{bmatrix} x_{51} & x_{52} & x_{53} & x_{54} \\ x_{61} & x_{62} & x_{63} & x_{64} \\ x_{71} & x_{72} & x_{73} & x_{74} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 7 & 5 & 3 & 2 \\ 3 & 5 & 10 & 15 \end{bmatrix} = [x_{ij}]$$

Putem calcula acum, folosind formula (6—38), valorile F_j ($j = 1, 2, 3, 4$) ale funcţiei obiectiv (6—42):

$$[F_1 \ F_2 \ F_3 \ F_4] = [c_5 \ c_6 \ c_7] \cdot [x_{ij}] = [0 \ 0 \ 0 \ 0],$$

Scanned with OKEN Scanner

deoarece $c_5 = c_6 = c_7 = 0$. Cu aceste valori putem calcula valorile $c_j - F_j$ $(j = 1, 2, 3, 4)$:

$$c_1 - F_1 = 60 - 0 = 60,$$
$$c_2 - F_2 = 60 - 0 = 60,$$
$$c_3 - F_3 = 90 - 0 = 90,$$
$$c_4 - F_4 = 90 - 0 = 90.$$

Urmează să trecem la o nouă bază; pentru aceasta, analizînd valorile $c_j - F_j$ $(j = 1, 2, 3, 4)$ de mai sus, rezultă că trebuie să alegem între intrarea în noua bază a coloanelor (*3*) sau (*4*) pentru care avem aceeași valoare (90), cea mai mare din cele de mai sus. Să presupunem, mai întîi, că introducem în noua bază vectorul $\mathbf{A}_3$ și pentru a vedea ce coloană va părăsi baza trebuie să calculăm raporturile $x_i^{(1)}/x_{i3}$:

$$\frac{x_5^{(1)}}{x_{53}} = \frac{15}{1} = 15, \quad \frac{x_6^{(1)}}{x_{63}} = \frac{1}{3} \text{ și } \frac{x_7^{(1)}}{x_{73}} = \frac{1}{10}.$$

Deoarece $\min_i x_i^{(1)}/x_{i3} = x_7^{(1)}/x^{73}$, urmează că trebuie să scoatem din bază vectorul $\mathbf{A}_7$. Să considerăm, în continuare, că am fi ales coloana (*4*); rezultă pentru acest caz:

$$\frac{x_5^{(1)}}{x_{54}} = \frac{15}{1} = 15, \quad \frac{x_6^{(1)}}{x_{64}} = \frac{1}{2} = 0{,}5 \text{ și } \frac{x_7^{(1)}}{x_{74}} = \frac{1}{15} = 0{,}066$$

și deoarece $\min_i x_i^{(1)}/x_{i4} = x_7^{(1)}/x_{74}$ reiese că și în acest caz trebuie să scoatem din bază tot pe $\mathbf{A}_7$. Atunci, pentru a putea alege între coloanele (*3*) și (*4*), vom aplica formula (6—40), adică:

$$F^{(2)} = F^{(1)} + \theta_0(c_j - F_j),$$

unde, în acest caz, $c_j - F_j = 90$ și $F^{(1)} = 0$; va trebui, deci, să alegem acel $\theta_0 = \min_i x_i/x_{ij}$ — în cazul nostru 1/10, pentru coloana (*3*) sau 1/15 pentru coloana (*4*) — care face ca valoarea funcției obiectiv să crească cel mai mult. Prin urmare, vom alege cel mai mare θ_0, adică 1/10, pentru care $F^{(2)} = 0 + \frac{1}{10} \cdot 90 = 9$, deci crește cel mai mult. Rezultă că în noua bază va intra $\mathbf{A}_3$, pentru care F crește de la valoarea zero (corespunzător bazei inițiale) la valoarea 9.

Introducînd vectorul A_3 în bază, în locul vectorului $\mathbf{A}_7$, noua soluție $\mathbf{X}^{(2)}$ este:*

$$x_1^{(2)} = 0, \; x_2^{(2)} = 0,$$
$$x_3^{(2)} = \theta_0 = \frac{1}{10}, \; x_4^{(2)} = 0,$$
$$x_5^{(2)} = x_5^{(1)} - \theta_0 \, x_{53} = 15 - \frac{1}{10} \cdot 1 = \frac{149}{10},$$
$$x_6^{(2)} = x_6^{(1)} - \theta_0 \, x_{63} = 1 - \frac{1}{10} \cdot 3 = \frac{7}{10} \text{ și } x_7^{(2)} = 0,$$

* Fiind vorba de a doua soluție, din iterația doi, o notăm cu (2) plasat ca exponent.

Scanned with OKEN Scanner

care verifică sistemul dat (6—43) sau (6—45), adică:

$$\begin{matrix}\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 0 & 0\\ 7 & 5 & 3 & 2 & 0 & 1 & 0\\ 3 & 5 & 10 & 15 & 0 & 0 & 1\end{bmatrix}\\ (1)\ (2)\ (3)\ (4)\ (5)\ (6)\ (7)\end{matrix}\cdot\begin{bmatrix}0\\0\\1/10\\0\\149/10\\7/10\\0\end{bmatrix}=\begin{bmatrix}15\\1\\1\end{bmatrix}$$

şi face ca funcţia obiectiv să aibă valoarea $F^{(2)} = 60 \cdot 0 + 60 \cdot 0 + 90 \cdot 1/10 + 90 \cdot 0 + 0 \cdot 149/10 + 0 \cdot 7/10 + 0 \cdot 0 = 9$.

Reluăm calculul, deci efectuăm a doua iteraţie, în care baza are indicii (5), (6), (3); ea se obţine din baza precedentă, prin înlocuirea coloanei (7) cu (3).

Astfel, aplicînd din nou relaţiile (6—37), vom obţine:

$$\begin{matrix}\begin{matrix}\mathbf{A}_j\\ \begin{bmatrix}1 & 1 & 0 & 1\\ 7 & 5 & 0 & 2\\ 3 & 5 & 1 & 15\end{bmatrix}\\ (1)\ (2)\ (7)\ (4)\end{matrix}=\begin{matrix}\mathbf{A}_i\\ \begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 3\\ 0 & 0 & 10\end{bmatrix}\\ (5)\ (6)\ (3)\end{matrix}\cdot\begin{bmatrix}x_{51} & x_{52} & x_{57} & x_{54}\\ x_{61} & x_{62} & x_{67} & x_{64}\\ x_{31} & x_{32} & x_{37} & x_{34}\end{bmatrix}\end{matrix}\qquad (6—46)$$

şi inversînd matricea a doua obţinem:

$$\begin{bmatrix}x_{51} & x_{52} & x_{57} & x_{54}\\ x_{61} & x_{62} & x_{67} & x_{64}\\ x_{31} & x_{32} & x_{37} & x_{34}\end{bmatrix}=\begin{bmatrix}1 & 0 & -1/10\\ 0 & 1 & -3/10\\ 0 & 0 & 1/10\end{bmatrix}\cdot\begin{bmatrix}1 & 1 & 0 & 1\\ 7 & 5 & 0 & 2\\ 3 & 5 & 1 & 15\end{bmatrix}=$$

$$=\begin{bmatrix}7/10 & 5/10-1/10- & 5/10\\ 61/10 & 35/10-3/10- & 25/10\\ 3/10 & 5/10 & 1/10 & 15/10\end{bmatrix}=[x_{ij}].$$

Calculăm apoi, valorile funcţiei obiectiv F_j $(j = 1, 2, 7, 4)$, adică:

$$[F_1\ F_2\ F_7\ F_4] = [c_5\ c_6\ c_3]\cdot[x_{ij}] = [27\ 45\ 9\ 135],$$

cu care determinăm valorile $c_j - F_j$ şi anume:

$$c_1 - F_1 = 60 - 27 = 33,$$
$$c_2 - F_2 = 60 - 45 = 15,$$
$$c_7 - F_7 = 0 - 9 = -9,$$
$$c_4 - F_4 = 90 - 135 = -45,$$

de unde rezultă că în următoarea iteraţie vom introduce în bază pe $\mathbf{A}_1$, deoarece $c_j - F_j$ este maxim pentru $j = 1$. Apoi calculăm raportul x_i/x_{i1}:

$$\frac{x_3^{(2)}}{x_{31}} = \frac{1/10}{3/10} = \frac{1}{3},\quad \frac{x_5^{(2)}}{x_{51}} = \frac{149/10}{7/10} = \frac{149}{7}\quad \text{şi}\quad \frac{x_6^{(2)}}{x_{61}} = \frac{7/10}{61/10} = \frac{7}{61},$$

Scanned with OKEN Scanner

de unde rezultă că în iterația următoare vom scoate din bază vectorul $\mathbf{A}_6$, deoarece x_i/x_{i1} este minim pentru $i = 6$ (deci $\theta_0 = x_6^{(2)}/x_{61} = 7/61$).

Noua soluție de bază, cu care vom efectua calculele în iterația a treia, este $\mathbf{X}^{(3)}$:

$$x_1^{(3)} = \theta_0 = 7/61, \quad x_2^{(3)} = 0,$$

$$x_3^{(3)} = x_3^{(2)} - \theta_0 x_{31} = \frac{1}{10} - \frac{7}{61} \cdot \frac{3}{10} = \frac{4}{61}, \quad x_4^{(3)} = 0,$$

$$x_5^{(3)} = x_5^{(2)} - \theta_0 x_{51} = \frac{149}{10} - \frac{7}{61} \cdot \frac{7}{10} = \frac{904}{61},$$

$$x_6^{(3)} = 0 \text{ și } x_7^{(3)} = 0,$$

pentru care funcția obiectiv ia valoarea:

$$F^{(3)} = \sum_{i=1}^{n+m} c_i x_i = 60 \cdot \frac{7}{61} + 60 \cdot 0 + 90 \cdot \frac{4}{61} + 90 \cdot 0 + 0 \cdot 0 + 0 \cdot \frac{904}{61} + 0 \cdot 0 = \frac{780}{61},$$

sau

$$F^{(3)} = F^{(2)} + \theta_0(c_1 - F_1) = 9 + \frac{7}{61} \cdot 33 = \frac{780}{61},$$

iar sistemul (6—45) devine:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 7 & 5 & 3 & 2 & 0 & 1 & 0 \\ 3 & 5 & 10 & 15 & 0 & 0 & 1 \\ (1) & (2) & (3) & (4) & (5) & (6) & (7) \end{bmatrix} \cdot \begin{bmatrix} 7/61 \\ 0 \\ 4/61 \\ 0 \\ 904/61 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 15 \\ 1 \\ 1 \end{bmatrix}.$$

Reluăm calculul, în cea de a treia iterație în care baza este formată din coloanele (5), (1) și (3) ale matricei $\mathbf{A}$. Vom obține succesiv:

$$\begin{matrix} & \mathbf{A}_j & & & & \mathbf{A}_i & & \\ \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 5 & 0 & 2 \\ 0 & 5 & 1 & 15 \\ (6) & (2) & (7) & (4) \end{bmatrix} & = & \begin{bmatrix} 1 & 1 & 1 \\ 0 & 7 & 3 \\ 0 & 3 & 10 \\ (5) & (1) & (3) \end{bmatrix} & \cdot & \begin{bmatrix} x_{56} & x_{52} & x_{57} & x_{54} \\ x_{16} & x_{12} & x_{17} & x_{14} \\ x_{36} & x_{32} & x_{37} & x_{34} \end{bmatrix}; & (6-47) \end{matrix}$$

$$\begin{bmatrix} x_{56} & x_{52} & x_{57} & x_{54} \\ x_{16} & x_{12} & x_{17} & x_{14} \\ x_{36} & x_{32} & x_{37} & x_{34} \end{bmatrix} = \begin{bmatrix} 1 & -7/61 & -4/61 \\ 0 & 10/61 & -3/61 \\ 0 & -3/61 & 7/61 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 5 & 0 & 2 \\ 0 & 5 & 1 & 15 \end{bmatrix} =$$

$$= \begin{bmatrix} -7/61 & 6/61 & -4/61 & -13/61 \\ 10/61 & 35/61 & -3/61 & -25/61 \\ -3/61 & 20/61 & 7/61 & 99/61 \end{bmatrix} = [x_{ij}];$$

Scanned with OKEN Scanner

$$[F_6\ F_2\ F_7\ F_4] = [c_5\ c_1\ c_3] \cdot [x] = [0\ 60\ 90] \cdot [x_{ij}] =$$
$$= [330/61\quad 3900/61\quad 450/61\quad 7410/61];$$
$$c_6 - F_6 = 0 - 330/61 = -330/61,$$
$$c_2 - F_2 = 60 - 3900/61 = -240/61,$$
$$c_7 - F_7 = 0 - 450/61 = -450/61,$$
$$c_4 - F_4 = 90 - 7410/61 = -1920/61.$$

Deoarece, pentru orice $j(j = 6, 2, 7, 4)$, $(c_j - F_j) < 0$ calculul nu mai poate fi continuat şi soluţia optimală va fi $\mathbf{X} = \mathbf{X}^{(3)}$, adică:

$$x_1 = \frac{7}{61} = 0{,}1148;\quad x_2 = 0;\quad x_3 = \frac{4}{61} = 0{,}0656;\quad x_4 = 0;\quad x_5 = \frac{904}{61} = 14{,}8197;$$

$x_6 = 0$ şi $x_7 = 0$, pentru care funcţia obiectiv are $max F = F^{(3)} = 780/61 = 12{,}79$.

Ex. 6—8. Să se găsească, prin metoda simplex, minimul funcţiei $F = 15x_1 + 33x_2$ cu restricţiile:

$$\begin{aligned} 3x_1 + 2x_2 &\geqslant 6,\\ 6x_1 + x_2 &\geqslant 6,\\ x_2 &\geqslant 1, \end{aligned}$$

şi respectarea condiţiilor de nenegativitate $x_j \geqslant 0$ $(j = 1, 2)$.

Modelul acestei probleme, cu variabilele de ecart, este următorul:

$$F = 15x_1 + 33x_2 + 0 \cdot x_3 + 0 \cdot x_4 + 0 \cdot x_5 = min$$

$$\begin{cases} 3x_1 + 2x_2 - x_3 = 6\\ 6x_1 + x_2 - x_4 = 6\\ x_2 - x_5 = 1 \end{cases}$$

$$x_j \geqslant 0 \qquad j = 1, 2, \ldots, 5$$

În acest caz:

$m = 3$ ecuaţii şi $n = 2$ variabile;

$$\mathbf{A} = \begin{bmatrix} 3 & 2 & -1 & 0 & 0\\ 6 & 1 & 0 & -1 & 0\\ 0 & 1 & 0 & 0 & -1 \end{bmatrix}, \text{ care are } n_c = m + n = 5 \text{ coloane;}$$
$$(1)\ (2)\ (3)\ (4)\ (5)$$

$$\mathbf{B} = \begin{bmatrix} 6\\ 6\\ 1 \end{bmatrix},\ \mathbf{C} = [15\ 33\ 0\ 0\ 0] \text{ şi } \mathbf{X} = \begin{bmatrix} x_1\\ x_2\\ x_3\\ x_4\\ x_5 \end{bmatrix}.$$

Alegem, în mod arbitrar, o soluţie de bază iniţială; de exemplu:

$$x_1^{(1)} = 2/3,\quad x_2^{(1)} = 2,\quad x_3^{(1)} = 0,\quad x_4^{(1)} = 0 \text{ şi } x_5^{(1)} = 1,$$

pentru care funcţia obiectiv ia valoarea $F^{(1)} = 76$.

Scanned with OKEN Scanner

Aceasta înseamnă că baza $\mathbf{A}_i$ este formată din coloanele (1), (2) și (5) ale matricei $\mathbf{A}$ și se va putea scrie:

$$\begin{bmatrix} x_{13} & x_{14} \\ x_{23} & x_{24} \\ x_{53} & x_{54} \end{bmatrix} = \underbrace{\begin{bmatrix} 3 & 2 & 0 \\ 6 & 1 & 0 \\ 0 & 1 & -1 \end{bmatrix}^{-1}}_{\mathbf{A}_i^{-1}} \cdot \underbrace{\begin{bmatrix} -1 & 0 \\ 0 & -1 \\ 0 & 0 \end{bmatrix}}_{\mathbf{A}_j} = \begin{bmatrix} 1/9 & -2/9 \\ -2/3 & 1/3 \\ -2/3 & 1/3 \end{bmatrix} = [x_{ij}];$$

$$[F_3\ F_4] = [c_1\ c_2\ c_5] \cdot [x_{ij}] = [15\ 33\ 0] \cdot \begin{bmatrix} 1/9 & -2/9 \\ -2/3 & 1/3 \\ -2/3 & 1/3 \end{bmatrix} = [-61/3\ \ 23/3];$$

$$c_3 - F_3 = 0 - (-61/3) = 61/3,$$
$$c_4 - F_4 = 0 - 23/3 = -23/3.$$

Deoarece $\min_j (c_j - F_j) = -23/3$ pentru $j = 4$, vom introduce în următoarea bază coloana (4) din $\mathbf{A}$. Pentru a vedea ce coloană va părăsi baza, calculăm rapoartele $\theta_i = x_i^{(1)}/x_{i4}$:

$$\theta_1 = \frac{x_1^{(1)}}{x_{14}} = \frac{2/3}{-2/9} = -3;\quad \theta_2 = \frac{x_2^{(1)}}{x_{24}} = 6 \text{ și } \theta_5 = \frac{x_5^{(1)}}{x_{54}} = 3 = \theta_0;$$

deoarece $\min_i \{(\theta_i \geqslant 0)\} = 3$ pentru $i = 5$, din bază vom scoate coloana (5), pe care o vom înlocui cu (4).

Noua soluție de bază, în iterația a doua, va fi:

$$x_1^{(2)} = x_1^{(1)} - \theta_0 x_{14} = \frac{2}{3} - 3 \cdot \left(-\frac{2}{9}\right) = \frac{4}{3},$$

$$x_2^{(2)} = x_2^{(1)} - \theta_0 x_{24} = 1, \qquad x_3^{(2)} = x_3^{(1)} = 0,$$

$$x_4^{(2)} = \theta_0 = 3 \quad \text{și} \quad x_5^{(2)} = x_4^{(1)} = 0,$$

pentru care funcția obiectiv are valoarea:

$$F^{(2)} = \sum_{i=1}^{n_c} c_i \cdot x_i^{(1)} = 15 \cdot \frac{4}{3} + 33 \cdot 1 + 0 \cdot 0 + 0 \cdot 3 + 0 \cdot 0 = 53,$$

valoare ce se poate determina (pentru verificare!) și prin:

$$F^{(2)} = F^{(1)} + \theta_0(c_4 - F_4) = 76 + 3\left(0 - \frac{23}{3}\right) = 53.$$

Urmează cea de a doua iterație:

$$\begin{bmatrix} x_{13} & x_{15} \\ x_{23} & x_{25} \\ x_{43} & x_{45} \end{bmatrix} = \underbrace{\begin{bmatrix} 3 & 2 & 0 \\ 6 & 1 & -1 \\ 0 & 1 & 0 \end{bmatrix}^{-1}}_{\mathbf{A}_i^{-1}} \cdot \underbrace{\begin{bmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{bmatrix}}_{\mathbf{A}_j} = \begin{bmatrix} -\frac{1}{3} & \frac{2}{3} \\ 0 & -1 \\ -2 & 3 \end{bmatrix} = [x_{ij}];$$

Scanned with OKEN Scanner

$$[F_3\ F_5] = [c_1\ c_2\ c_4] \cdot [x_{ij}] = [15\ 33\ 0] \cdot \begin{bmatrix} -\frac{1}{3} & \frac{2}{3} \\ 0 & -1 \\ -2 & 3 \end{bmatrix} = [-5\ -23];$$

$$c_3 - F_3 = 0 - (-5) = 5 \quad \text{și} \quad c_5 - F_5 = 0 - (-23) = 23.$$

Deoarece nu există nici un $(c_j - F_j) < 0$ atunci calculul se oprește aici, soluția optimală fiind:

$$x_1 = 4/3, \qquad x_2 = 1, \qquad x_3 = 0, \qquad x_4 = 3 \quad \text{și} \quad x_5 = 0,$$

pentru care $F = 53$ reprezintă minimul căutat.

După cum rezultă din acest exemplu, o problemă de minim se rezolvă prin metoda simplex în același mod ca o problemă de maxim, cu deosebirile:

— se caută *min* $(c_j - F_j)$ care trebuie să fie negativ pentru orice j sau *max* $(F_j - c_j)$ care trebuie să fie pozitiv pentru orice j;

— calculul iterativ încetează atunci cînd nu mai există nici un $(c_j - F_j) < 0$ sau nici un $(F_j - c_j) > 0$.

Un algoritm și program de calcul pentru aplicarea metodei simplex va fi prezentat în continuare. Mai înainte, precizăm însă că actualmente există numeroase procedee de utilizare a metodei simplex: unele au rezultat din necesitatea de a face aplicabilă metoda simplex prin calcule „manuale" (ca, de exemplu, cunoscutul „procedeu al tabelelor"), altele — cele mai multe — au rezultat din necesitatea rezolvării programelor cu mari dimensiuni prin folosirea eficientă a sistemelor de calcul automat (cu micșorarea timpului-calculator prin reducerea numărului de iterații, sau cu micșorarea volumului necesar de memorie operativă).

Avînd în vedere scopul acestei lucrări, nu ne vom opri asupra metodelor „manuale", mai ales că aplicarea lor este limitată la programele de mici dimensiuni (cel mult cu zece variabile), în timp ce în practică, adesea, programele au astfel de dimensiuni (zeci și sute de variabile) încît rezolvarea lor nu se poate face decît prin utilizarea sistemelor de calcul automat. De altfel, dezvoltarea pe care o are în prezent optimizarea prin programare matematică se datorește în principal tehnicilor de calcul automat.

Pentru calculul automat au fost puși la punct numeroși algoritmi, majoritatea fiind de fapt variante ale metodei simplex ca: metoda pivotării, metoda descompunerii (Dantzig-Wolfe), metoda Kontorovici, metoda gradientului și multe altele. Nici asupra acestor metode, și așa destul de diverse, nu vom insista, deoarece majoritatea sistemelor de calcul automat dispun de pachete de programe specifice pentru optimizări prin programe liniare, simplu de folosit de către orice utilizator, prin proceduri pe care le vom prezenta în detaliu la finele acestui capitol.

Pentru a permite însă cititorului să înțeleagă cît mai bine principiul metodei simplex (element esențial pentru oricare utilizator ce are de formulat o problemă de programare liniară și care dorește să folosească corect un pachet de programe pentru optimizări), vom prezenta în continuare —

Scanned with OKEN Scanner

cu titlu de exerciţiu — un algoritm simplex şi programul sursă FORTRAN aferent.

Algoritmul respectă întocmai metoda simplex prezentată anterior şi se referă numai la programele cu maximizarea funcţiei obiectiv. Evident, prin cîteva simple modificări, el poate fi adaptat — chiar de către cititor, ca exerciţiu — şi pentru problemele de minim; singurele modificări sînt: la transformarea bazei se introduce acea coloană j din matricea **A** pentru care $(c_j - F_j) < 0$ să fie cea mai mare posibilă în valoare absolută, iar în loc să se caute $min\ F$ se va determina $max\ (-F)$.

Acest algoritm este prezentat, prin termenii specifici prelucrării datelor, în schema de calcul din figura 6.2.

Datele problemei (datele de „intrare") sînt următoarele: numărul de restricţii (m); numărul de variabile (n); matricea **A**, formată din coeficienţii a_{ij} ($i = 1, 2, \ldots, n; j = 1, 2, \ldots, m + n$) ai sistemului de m ecuaţii cu $n_c = m + n$ necunoscute, obţinut prin completarea restricţiilor cu cele m variabile de ecart x_{n+i} ($i = 1, 2, \ldots, m$); matricea **B**, formată din termenii liberi b_i ($i = 1, 2, \ldots, m$) ai restricţiilor; matricea „linie" **C**, formată din coeficienţii c_j ($j = 1, 2, \ldots, n_c$) ai funcţiei obiectiv, din care cei corespunzători variabilelor de ecart sînt nuli ($c_{n+i} = 0,\ i = 1, 2, \ldots m$); valorile atribuite iniţial elementelor soluţiei de bază, notate (pentru o mai directă corespondenţă cu numele de variabile utilizate în programul FORTRAN aferent) cu sb_j ($j = 1, 2, \ldots, n_c$); indicii coloanelor din matricea **A** care se introduc iniţial în baza $\mathbf{A}_j$, notaţi cu ib_i ($i = 1, 2, \ldots, m$) şi indicii coloanelor matricei **A** care nu se introduc în baza $\mathbf{A}_i$ şi formează matricea $\mathbf{A}_j$ (v. relaţia 6—37 şi, spre exemplificare, relaţia 6—45), notaţi în schema din figura 6.2, cu inb_k ($k = 1, 2, \ldots, n$).

În legătură cu aceste date de „intrare", singura dificultate (mai evidentă în cazul problemelor cu valori, $m, n > 6$) constă în alegerea iniţială a soluţiei de bază sb_j şi — implicit — a şirurilor ib_j şi inb_k($i = 1,2, \ldots$ $\ldots, m;\ j = 1, 2, \ldots, n_c;\ k = 1, 2, \ldots, n$), asupra căreia vom insita puţin.

Soluţia iniţială de bază poate fi aleasă prin diferite procedee; cel pe care îl prezentăm este cunoscut sub denumirea de „metoda celor două faze" [17], deoarece se realizează prin două etape („faze") de calcul.

Dacă ne referim la următorul model (mai general) al unei probleme de programare liniară şi anume:

$$\begin{cases} \sum_{j=1}^{n} a_{ij}\, x_j = b_i, & i \in N_1 \\ \sum_{j=1}^{n} a_{ij} x_j \geqslant b_i, & i \in N_2 \\ \sum_{j=1}^{n} a_{ij} x_j \leqslant b_i, & i \in N_3 \\ x_j \geqslant 0,\ j \in \{1, 2, \ldots, n\} \end{cases} \tag{1}$$

$$F = \sum_{j=1}^{n} c_j x_j = max \text{ cu } b_i \geqslant 0,\ i = 1, 2, \ldots, m, \tag{2}$$

Scanned with OKEN Scanner

atunci sistemul de restricţii (1) poate fi scris, prin introducerea variabilelor de ecart x_i^e şi a unor variabile artificiale x_i^a, sub forma:

$$\begin{cases} \sum_{j=1}^{n} a_{ij} x_j + x_i^a = b_i, & i \in N_1 \\ \sum_{j=1}^{n} a_{ij} x_j - x_i^e + x_i^a = b_i, & i \in N_2 \\ \sum_{j=1}^{n} a_{ij} x_j + x_i^e = b_i, & i \in N_3 \\ x_j \geqslant 0, & j \in \{1, 2, \ldots, n\} \\ x_i^a \geqslant 0, & i \in N_1 \cup N_2;\ x_i^e \geqslant 0,\ i \in N_2 \cup N_3 \end{cases} \tag{3}$$

Poliedrul P_a definit de sistemul (3) are, evident, un vîrf admisibil:

$$x_i^a = b_i,\ i \in N_1 \cup N_2,$$

$$x_i^e = 0,\ i \in N_2,$$

$$x_i^e = b,\ i \in N_3,$$

$$x_j = 0,\ j = 1, 2, \ldots, n.$$

poate constitui o soluţie de bază iniţială (cu care să începem calculul). Cu aceasta, rezolvarea problemei (1), (2) se poate realiza prin două faze distincte.

Faza 1, în care se rezolvă problema $\varepsilon = \sum_{i \in N_1 \cup N_3} x_i^a = min$, în condiţiile (3). Pentru aceasta, se aplică algoritmul simplex pînă se ajunge în una din următoarele situaţii:

situaţia 1a în care $\varepsilon = 0$ şi nici o variabilă artificială nu face parte din bază (un vîrf al lui P_a este un vîrf al poliedrului iniţial P). De aici se trece la faza 2, situaţia 2a;

situaţia 1b în care $\varepsilon = 0$ însă o variabilă artificială face parte din bază (vîrful lui P_a este un vîrf al lui P). De aici se trece la faza 2, situaţia 2b;

situaţia 1c în care $min\ \varepsilon > 0$, adică poliedrul P este vid (ceea ce înseamnă că restricţiile sînt contradictorii).

Faza 2. Se reia funcţia obiectiv F din (2), existînd (în funcţie de situaţiile fazei precedente) două posibilităţi de efectuare a calculului:

situaţia 2a, atunci cînd am ajuns în *1a*, pentru care se aplică algoritmul simplex poliedrului iniţial P şi funcţiei obiectiv F (pînă la maximizarea ei);

situaţia 2b, atunci cînd am ajuns în *1b*, pentru care se aplică algoritmul simplex poliedrului P_a (restrîns la muchiile lui, care conservă valoarea nulă a lui ε). Dacă am găsit un vîrf al lui P_a se caută, mai întîi, toate muchiile care nu măresc pe ε şi — dintre ele — se alege aceea care măreşte pe F. Astfel vîrful optim va aparţine şi lui P.

În cazul schemei din figura 6.2, dacă s-a ales iniţial o soluţie de bază admisibilă, problema de programare liniară se rezolvă iterativ („contorizarea" iteraţiilor se face cu ajutorul variabilei *ni*, cărei iniţial i se atribuie valoarea zero), în fiecare iteraţie efectuîndu-se următoarele operaţii:

— constituirea matricelor $\mathbf{A}_i$ (cu coloanele ib_i, $i = 1, 2, \ldots, m$, din $\mathbf{A}$) şi $\mathbf{A}_j$ (cu coloanele inb_k, $k = 1, 2, \ldots, n$, din $\mathbf{A}$), operaţii reprezentate în figura 6.2 prin porţiunea de schemă de la *a* la *b*;

— inversarea matricei $\mathbf{A}_i$ şi efectuarea produsului matricelor $\mathbf{A}_i^{-1} \cdot \mathbf{A}_j$, din care rezultă matricea $\mathbf{X}$, cu elementele x_{ij} ($i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$), reprezentate prin porţiunea de schemă $c \div d$;

Scanned with OKEN Scanner

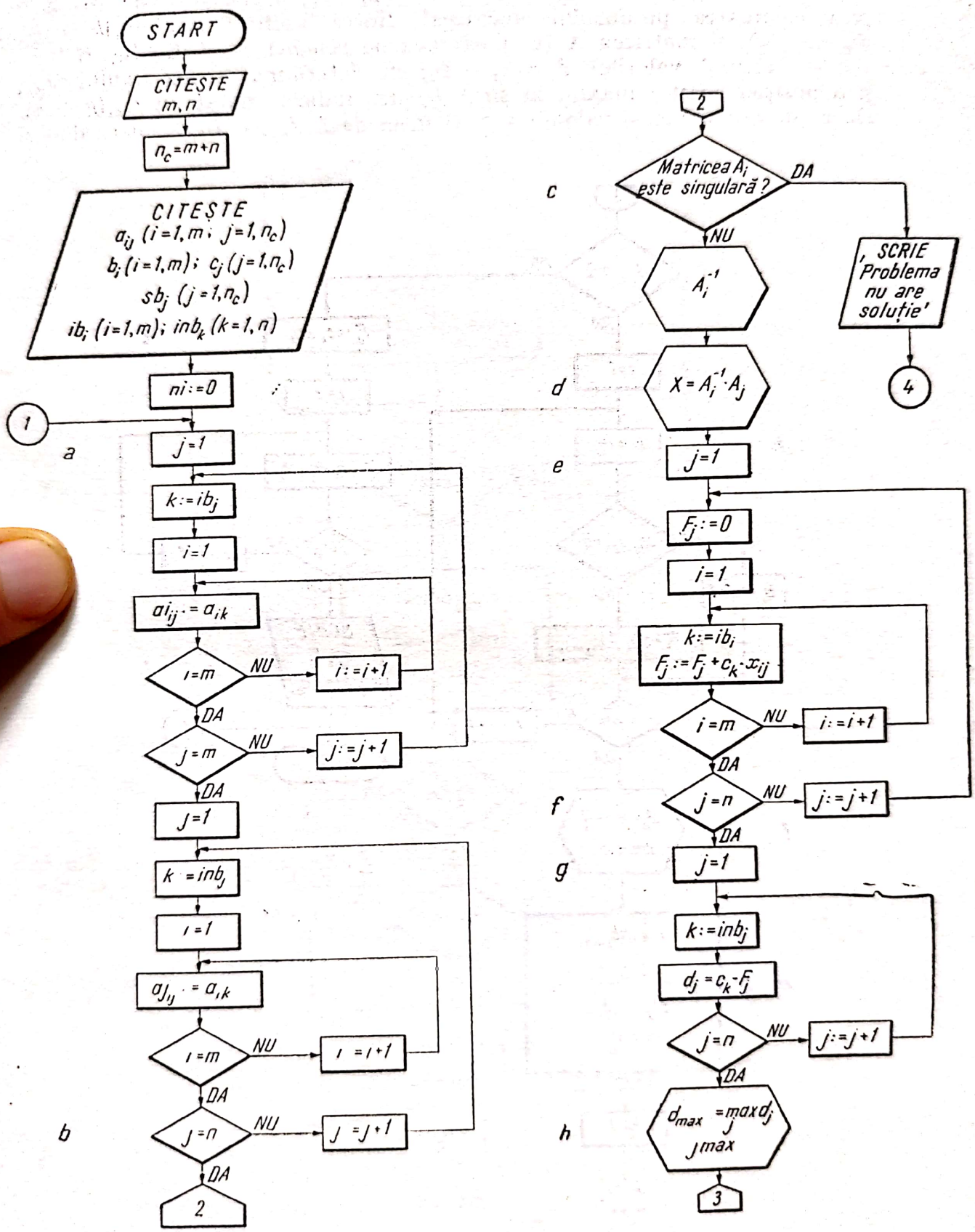


Fig. 6.2

Scanned with OKEN Scanner

— determinarea valorilor F_j, $j = 1, 2, \ldots, n$, definite de (6—38), prin efectuarea produsului matriceal dintre vectorul linie $[c_k]k = ib_1, ib_2, \ldots, ib_m$ și matricea **A** (v. porțiunea de schemă $e \div f$ din fig. 6.2);

— calculul valorilor $d_j = c_j - F_j$, cu determinarea maximului d_{max} și a poziției acestui maxim în șirul d_j, prin indicele notat cu $j_{max}(g \div h)$. Dacă nu există nici o valoare $d_j > 0$ (deci dacă $d_{max} < 0$), atunci calculul

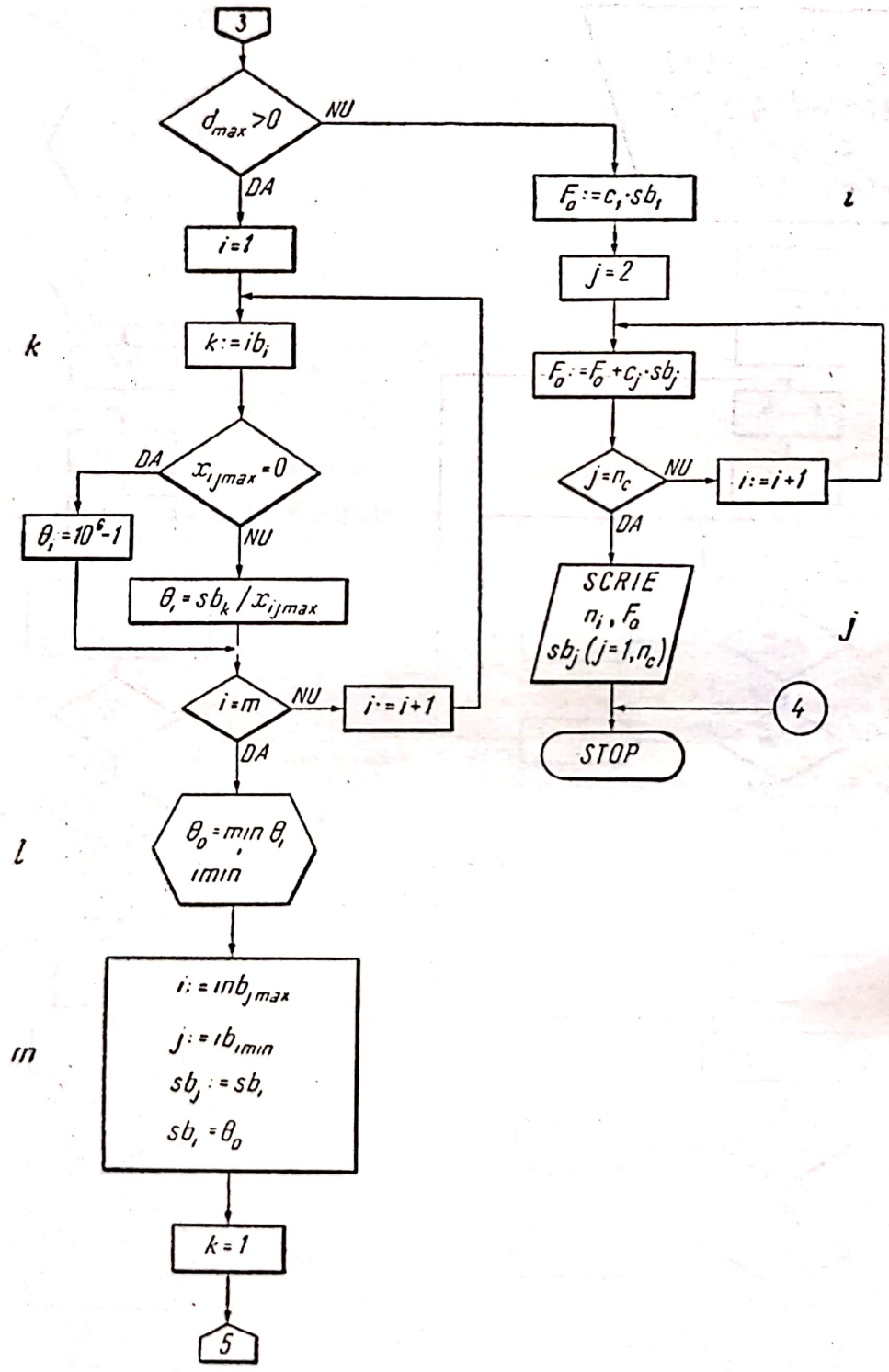


Fig. 6.2

Scanned with OKEN Scanner

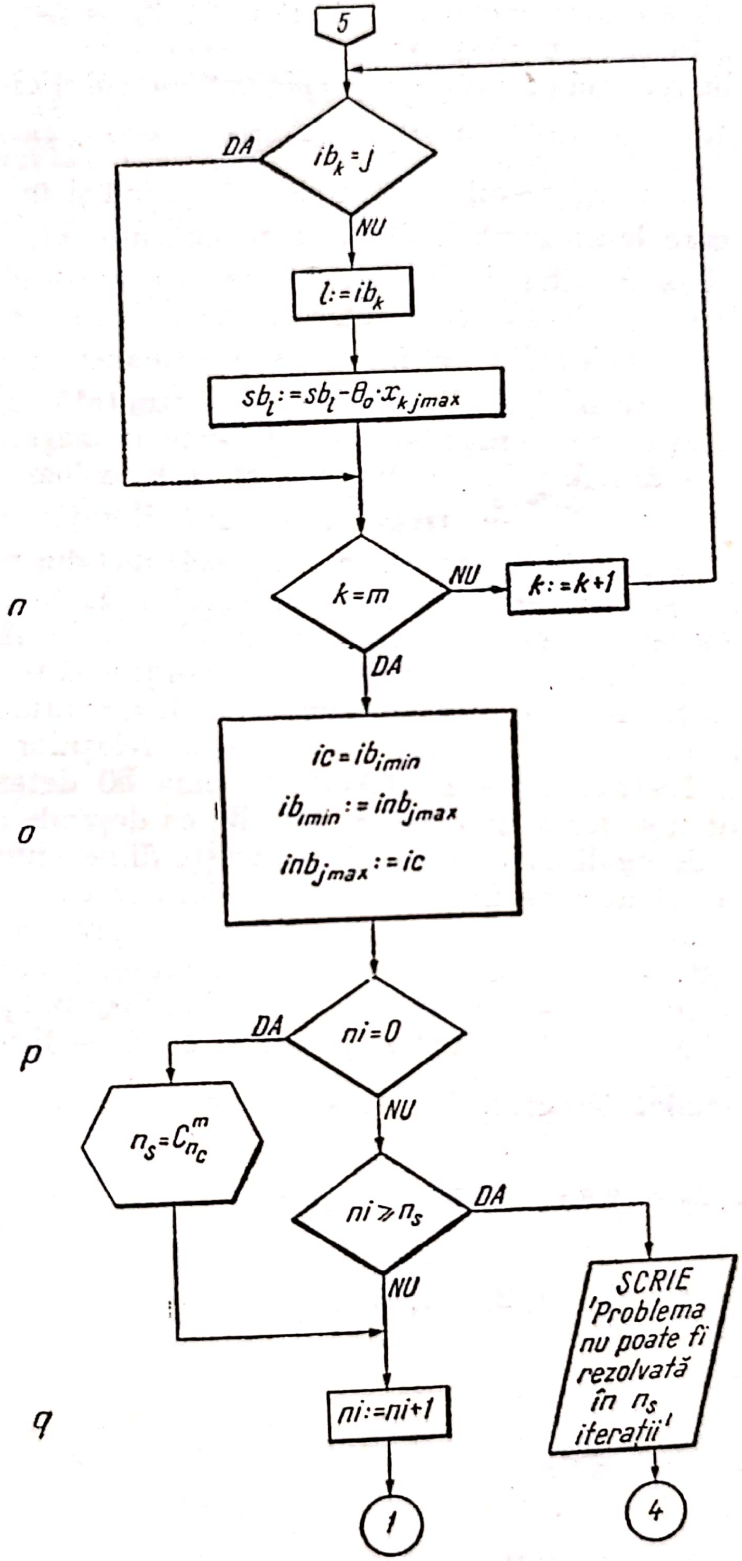


Fig. 6.2

Scanned with OKEN Scanner

iterativ ia sfîrşit şi valorile pe care le au acum sb_j constituie soluţia optimală, cu care se determină $F_0 = max\, F(i \div j)$;

— dacă există un $d_j > 0$ (care poate fi chiar d_{max}), atunci calculul iterativ continuă cu determinarea valorilor θ_i, $\theta_0 = min\ \theta_i$ şi a poziţiei *(imin)* a lui θ_0 în şirul θ_i $(k \div l)$;

— determinarea noilor valori ale elementelor soluţiei de bază:

$sb_{inb_{jmax}}^{(ni+1)} = \theta_0$, $sb_{ib_{imin}}^{(ni+1)} = sb_{ib_{imin}}^{(ni)}$ şi $sb_{ib_k}^{(ni+1)} = sb_{ib_k}^{(ni)} - \theta_0^{(ni)} \cdot x_{kjmax}^{(ni)}$ pentru $k = 1, 2, \ldots, m$ însă $k \neq ib_{imin}$, restul elementelor rămînînd şi în iteraţia $ni + 1$ cu valorile pe care le-au avut în iteraţia precedentă ni, adică $sb_j^{(ni+1)} = sb_j^{(ni)}$ pentru indicii j diferiţi de cei de mai sus $(m \div n)$;

— determinarea coloanei care intră în noua bază, ce va fi utilizată în iteraţia $ni + 1$, adică $ib_{min}^{(ni+1)} = inb_{jmax}^{(ni)}$, şi a coloanei care iese din bază $inb_{jmax}^{(ni+1)} = ib_{imin}^{(ni)}$ (operatorul *o* din schema reprezentată în figura 6.2);

— dacă numărul de iteraţii ni nu depăşeşte numărul de matrice $\mathbf{A}_i$ care se pot forma din $\mathbf{A}$ prin combinarea celor n_c coloane luate cîte m, adică dacă $ni < n_s = C_{n_c}^{m}$, se trece la o nouă iteraţie $(ni + 1)$. Dacă $mi \geqslant n_s$ calculul este oprit (porţiunea $p \div q$ a schemei din figura 6.2).

În lista 6.1 este prezentat programul sursă redactat în FORTRAN care realizează întocmai algoritmul redat prin schema din figura 6.2. Acest program, care are mai mult un interes didactic, conţine şi toate comentariile necesare pentru înţelegerea şi utilizarea lui. Numele de variabile şi expresiile respectă întru totul notaţiile şi felul de scriere al relaţiilor din schema de calcul (fig. 6.2). Instrucţiunea DATA de la linia 80 determină formatul variabil FW pentru scrierea matricelor **A** şi **B**; ea depinde de dimensiunile problemei şi trebuie modificată în funcţie de aceste dimensiuni (în programul reprodus în lista 6.1 această instrucţiune se referă la cazul din *Ex. 6—1*).

Acest program a fost utilizat pentru rezolvarea problemelor din unele exemple prezentate pînă acum. Rezultatele obţinute, precedate de datele problemei şi de valorile alese iniţial, sînt redate în: lista 6.2 pentru *Ex.6—7*, lista 6.3 pentru *Ex. 6—1* (rezolvat şi prin *Ex. 6—6*) şi lista *6—4*, cazul *c*.

Programe duale. Programele:

$$\begin{cases} F = \sum_{j=1}^{n} c_j\, x_j = max \\ \sum_{j=1}^{n} a_{ij}\, x_j \leqslant b_i, \quad i = 1, 2, \ldots, m \end{cases} \qquad (6-48)$$

pe de o parte şi

$$\begin{cases} G = \sum_{i=1}^{m} b_i\, y_i = min. \\ \sum_{i=1}^{m} a_{ij}\, y_i \geqslant c_j, \quad j = 1, 2, \ldots, n \end{cases} \qquad (6-49)$$

pe de altă parte, sînt denumite programe duale. Ele au următoarele proprietăţi:

Scanned with OKEN Scanner

```
      C  LISTA 6.1
      C  ************************************************************
      C  *  PROGRAM FORTRAN PENTRU REZOLVAREA PROBLEMELOR DE PROGRAMARE  *
      C  *  LINIARA CU CEL MULT 20 DE VARIABILE SI MAXIMIZAREA FUNCTIEI  *
      C  *  OBIECTIV - METODA GENERALA SIMPLEX  (CRITERIUL LUI DANTZIG)  *
      C  ************************************************************
      C
      C  PARAMETRII DE INTRARE SINT:
      C     M       - NUMARUL DE RESTRICTII;
      C     N       - NUMARUL DE VARIABILE;
      C     A(I,J)  - ELEMENTELE MATRICEI A (COEFICIENTII SISTEMULUI DE RESTRICTII);
      C     B(I)    - ELEMENTELE MATRICEI B (TERMENII LIBERI AI RESTRICTIILOR);
      C     C(J)    - ELEMENTELE MATRICEI C (COEFICIENTII FUNCTIEI OBIECTIV);
      C     SB(J)   - ELEMENTELE SOLUTIEI DE BAZA (ALESE INITIAL);
      C     IB(I)   - INDICII COLOANELOR DIN MATRICEA A CARE INTRA IN BAZA;
      C     INB(K)  - INDICII COLOANELOR MATRICEI A CARE NU INTRA IN BAZA;
      C     INDICII DE MAI SUS POT LUA URMATOARELE VALORI:
      C           I DE LA 1 LA M CU PASUL 1,
      C           J DE LA 1 LA M+N CU PASUL 1,
      C           K DE LA 1 LA N CU PASUL 1,
      C     VALORILE ACESTOR PARAMETRI SE INTRODUC PRIN 7 TIPURI DE CARTELE CU
      C     URMATOAREA MACHETA:
      C     CARTELA              PARAMETRII          COLOANELE        FORMATUL
      C     -------              ----------          ---------        --------
      C  PRIMA (TIPUL 1):           M                 1 SI 2            XX
      C                             N                 3 SI 4            XX
      C
      C  URMATOARELE              A(I,J)              1 LA 8
      C  CEL MULT 40          (CITE 10 VALORI        .......          XXXX.XXX
      C   (TIPUL 2)          PE FIECARE CARTELA)     73 LA 80
      C   -COEFICIENTII RESTRICTIILOR A(I,J) SE PERFOREAZA COLOANA DUPA COLOANA-
      C
      C  URMATOARELE               B(I)               1 LA 8
      C  CEL MULT 2           (CITE 10 VALORI        .......          XXXX.XXX
      C   (TIPUL 3)          PE FIECARE CARTELA)     73 LA 80
      C
      C  URMATOARELE               C(J)               1 LA 8
      C  CEL MULT 2           (CITE 10 VALORI        .......          XXXX.XXX
      C   (TIPUL 4)          PE FIECARE CARTELA)     73 LA 80
      C
      C  URMATOARELE              SB(J)               1 LA 8
      C  CEL MULT 2           (CITE 10 VALORI        .......          XXX.XXXX
      C   (TIPUL 5)          PE FIECARE CARTELA)     73 LA 80
      C
      C PENULTIMA (TIP 6):        IB(I)          1-2, 3-4,...           XX
      C
      C  ULTIMA (TIP 7):          INB(K)         1-2, 3-4,...           XX
      C
      C  PARAMETRII DE IESIRE SINT:
      C     FO    - VALOAREA MAXIMA A FUNCTIEI OBIECTIV;
      C     SB(J) - VALORILE SOLUTIEI DE BAZA DUPA ULTIMA ITERATIE PENTRU CARE
      C             FUNCTIA OBIECTIV ARE VALOAREA MAXIMA - SOLUTIA OPTIMALA;
      C     NI    - NUMARUL DE ITERATII;
      C
      C  URMATOARELE VARIABILE DIN PROGRAM SINT UTILIZATE CA PARAMETRI
      C  INTERMEDIARI (DE CALCUL):
      C     NC      - NUMARUL DE COLOANE AL MATRICEI A;
      C     NS      - NUMARUL DE SOLUTII AI PROBLEMEI (DETERMINAT PRIN COMBI-
      C               NARI DE NC LUATE CITE M);
      C     AI      - TABLOUL MATRICEI ''BAZA'' -V. EXPRESIA (6-37);
      C     AJ      - TABLOUL COMPLEMENTILOR BAZEI AI IN A -V. EXPRESIA (6-37);
      C     AIM1    - MATRICEA INVERSA A BAZEI AI;
```

```
C       X         - MATRICEA PRODUSULUI MATRICELOR AIM1 SI AJ;
C       F(J)      - VALORI DETERMINATE CONFORM EXPRESIEI (6-38);
C       D(J)      - DIFERENTELE F(J)-F(J);
C       DMAX      - VALOAREA MAXIMA A DIFERENTELOR D(J);
C       JMAX      - INDICELE VALORII MAXIME A DIFERENTELOR D(J);
C       TETA(I)   - RAPOARTELE SB(K)/X(I,JMAX);
C       TETAO     - VALOAREA MINIMA A RAPOARTELOR TETA(I);
C       IMIN      - INDICELE VALORII MINIME A RAPOARTELOR TETA(I);
C       IC        - VARIABILA PENTRU COMUTAREA UNOR VALORI INTRE SIRURILE
C                   IB SI INB.
C
C  ALOCAREA MEMORIEI:
      DIMENSION A(20,20),B(20),C(20),SB(20),IB(20),INB(20),F(20),FW(4)
      COMMON M,N/MATRI/AIM1(20,20)/INV/AI(20,20),IER/PROD/AJ(20,20),X(20
     1,20)/MAX/D(20),DMAX,JMAX/MIN/TETA(20),TETAO,IMIN
C  INTRODUCEREA DATELOR:
      DATA LC,IM/105,108/
      DATA FW/4H(5(4,4HX,F8,4H,3)),4H    /
      READ (LC,50)M,N
      NC=M+N
      READ (LC,51) ((A(I,J),I=1,M),J=1,NC)
      READ (LC,51) (B(I),I=1,M)
      READ (LC,51) (C(J),J=1,NC)
      READ (LC,52) (SB(J),J=1,NC)
      READ (LC,53) (IB(I),I=1,M)
      READ (LC,53) (INB(K),K=1,N)
C  LISTAREA VALORILOR INTRODUSE:
      WRITE (IM,80)
      WRITE (IM,FW) ((A(I,J),J=1,NC),B(I),I=1,M)
      WRITE (IM,81) (C(J),J=1,NC)
      WRITE (IM,82) (SB(J),J=1,NC)
      WRITE (IM,83) (IB(I),I=1,M)
      WRITE (IM,84) (INB(K),K=1,N)
C  CALCULE SI DETERMINARI PRELIMINARE
      CALL COMBIN(M,NC,NS)
      NI=0
C  ALCATUIREA MATRICELOR AI SI AJ:
    1 DO 2 J=1,M
      K=IB(J)
      DO 2 I=1,M
    2 AI(I,J)=A(I,K)
      DO 3 J=1,N
      K=INB(J)
      DO 3 I=1,M
    3 AJ(I,J)=A(I,K)
C  INVERSAREA MATRICEI AI PRIN UTILIZAREA SUBRUTINEI 'INVERS' PENTRU CARE
C    PARAMETRUL IER SEMNALEAZA DACA AI ESTE NESINGULARA (PRIN VALOAREA IER=0)
C    SAU DACA ESTE SINGULARA (PRIN VALOAREA IER=-1); DACA IER=-1 SOLUTIA PRO-
C    BLEMEI NU ESTE POSIBILA; MATRICEA AIM1 ESTE INVERSA BAZEI AI:
      CALL INVERS
      IF(IER.EQ.-1) GO TO 12
      IF(IER.EQ.0) GO TO 4
      WRITE (IM,88)
      GO TO 14
C  EFECTUAREA PRODUSULUI MATRICELOR AIM1 CU AJ PRIN UTILIZAREA SUBRUTINEI
C    'PRODMA', MATRICEA X ESTE PRODUSUL AIM1 CU AJ:
    4 CALL PRODMA
C  DETERMINAREA VALORII DMAX:
      DO 5 J=1,N
      F(J)=0.
      DO 5 I=1,M
      K=IB(I)
```


Scanned with OKEN Scanner

```
      5 F(J)=F(J)+C(K)*X(I,J)
        DO 6 J=1,N
        K=INB(J)
      6 D(J)=C(K)-F(J)
        CALL MAXIM
C  VERIFICAREA POSIBILITATII DE CRESTERE A FUNCTIEI OBIECTIV:
        IF(DMAX) 10,10,7
C  CONTINUAREA CALCULULUI ITERATIV
C    APLICAREA CRITERIULUI ''DANTZIG'':
      7 DO 8 I=1,M
        K=IB(I)
        IF(X(I,JMAX)) 18,28,18
     28 TETA(I)=9999999.
        GO TO 8
     18 TETA(I)=SB(K)/X(I,JMAX)
      8 CONTINUE
        CALL MINIM
C     DETERMINAREA UNEI NOI SOLUTII DE BAZA:
        I=INB(JMAX)
        J=IB(IMIN)
        SB(J)=SB(I)
        SB(I)=TETA0
        DO 9 K=1,M
        IF(IB(K).EQ.J) GO TO 9
        L=IB(K)
        SB(L)=SB(L)-TETA0*X(K,JMAX)
      9 CONTINUE
C     DETERMINAREA INDICILOR CARE INTRA/IES IN/DIN NOUA BAZA:
        IC=IB(IMIN)
        IB(IMIN)=INB(JMAX)
        INB(JMAX)=IC
C     TRECEREA LA EXECUTAREA UNEI NOI INTERATII:
        IF(NI.GE.NS) GO TO 13
        NI=NI+1
        GO TO 1
C  EXTRAGEREA REZULTATELOR:
     10 FO=C(1)*SB(1)
        DO 11 J=2,NC
     11 FO=FO+C(J)*SB(J)
        WRITE (IM,85) NI,FO,(J,SB(J),J=1,NC)
        GO TO 14
     12 WRITE (IM,86) NI
        GO TO 14
     13 WRITE (IM,87) NI
     14 STOP
C  FORMATE DE INTRODUCERE A DATELOR:
     50 FORMAT(2I2)
     51 FORMAT(10F8.3)
     52 FORMAT(10F8.4)
     53 FORMAT(20I2)
C  FORMATE DE EXTRAGERE A REZULTATELOR:
     80 FORMAT('1',20X,'DATELE PROBLEMEI SINT URMATOARELE:'//'0',5X,'MATRI
       1CEA COEFICIENTILOR DIN SISTEMUL DE RESTRICTII COMPLETAT CU VARIABI
       2LELE DE ECART'/' ',17X,'(DE ULTIMA COLOANA SINT TERMENII LIBERI)')
     81 FORMAT('0',5X,'VALORILE COEFICIENTILOR DIN FUNCTIA OBIECTIV (IN OR
       1DINEA NATURALA A INDICILOR):'/' ',10(5X,F8.3))
     82 FORMAT('0',5X,'VALORILE ALESE INITIAL PENTRU ELEMENTELE SOLUTIEI D
       1E BAZA (IN ORDINEA NATURALA A INDICILOR):'/' ',10(5X,F8.4))
     83 FORMAT('0',5X,'COEFICIENTII COLOANELOR DIN MATRICEA A INTRODUSE IN
       1ITIAL IN BAZA:'/' ',20(3X,I2))
     84 FORMAT('0',5X,'COEFICIENTII COLOANELOR DIN MATRICEA A CARE INITIAL
       1 NU INTRA IN BAZA:'/' ',20(3X,I2))
     85 FORMAT('1',15X,'REZULTATELE FINALE, DUPA',1X,I2,' ITERATII, SINT U
       1RMATOARELE:'/' ',14X,57('-')/'0',10X,'VALOAREA MAXIMA A FUNCTIEI O
```


Scanned with OKEN Scanner

```
      2BIECTIV:',2X,F8.2///'0',15X,'SOLUTIA OPTIMALA'//' ',14X,18(1H-)//'0'/
     3(16X,'X(',I2,')=',F8.4))
  86 FORMAT('1',5X,'PROBLEMA NU ARE SOLUTIE, DEOARECE LA ITERATIA ',I2,
     1' MATRICEA BAZEI ESTE SINGULARA. STOP')
  87 FORMAT('1',5X,'PROBLEMA NU POATE FI REZOLVATA IN',1X,I3,1X,'ITERAT
     1II. STOP')
  88 FORMAT('1',10X,'MATRICEA BAZEI AI NU ESTE CORECT ALCATUITA. STOP')
      END
C
C  PROCEDURA EXTERNA PENTRU INVERSAREA MATRICELOR
      SUBROUTINE INVERS
      COMMON M,N/MATRI/AIM1(20,20)/INV/AI(20,20),IER
      EPS=.0000001
      DO 3 I=1,M
      DO 3 J=1,M
      IF(I-J) 2,1,2
    1 AIM1(I,J)=1.
      GO TO 3
    2 AIM1(I,J)=.0
    3 CONTINUE
      DEL=1.
      DO 14 K=1,M
      IF(K-M) 4,9,9
    4 IMAX=K
      AMAX=ABS(AI(K,K))
      KP1=K+1
      DO 6 I=KP1,M
      IF(AMAX-ABS(AI(I,K))) 5,6,6
    5 IMAX=I
      AMAX=ABS(AI(I,K))
    6 CONTINUE
      IF(IMAX-K) 7,9,7
    7 DO 8 J=1,M
      AITMP=AI(IMAX,J)
      AI(IMAX,J)=AI(K,J)
      AI(K,J)=AITMP
      AIM1T=AIM1(IMAX,J)
      AIM1(IMAX,J)=AIM1(K,J)
    8 AIM1(K,J)=AIM1T
      DEL=-DEL
    9 CONTINUE
      IF(ABS(AI(K,K))-EPS) 15,15,10
   10 DEL=AI(K,K)*DEL
      DIV=AI(K,K)
      DO 11 J=1,M
      AI(K,J)=AI(K,J)/DIV
   11 AIM1(K,J)=AIM1(K,J)/DIV
      DO 14 I=1,M
      AIMULT=AI(I,K)
      IF(I-K) 12,14,12
   12 DO 13 J=1,M
      AI(I,J)=AI(I,J)-AIMULT*AI(K,J)
   13 AIM1(I,J)=AIM1(I,J)-AIMULT*AIM1(K,J)
   14 CONTINUE
      IER=0
      GO TO 16
   15 IER=-1
   16 RETURN
      END
```


Scanned with OKEN Scanner

```
      C
      C  PROCEDURA EXTERNA PENTRU EFECTUAREA PRODUSULUI DINTRE DOUA MATRICE
            SUBROUTINE PRODMA
            COMMON M,N/MATRI/AIM1(20,20)/PROD/AJ(20,20),X(20,20)
            DO 1 I=1,M
            DO 1 K=1,N
            X(I,K)=0.
            DO 1 J=1,M
          1 X(I,K)=X(I,K)+AIM1(I,J)*AJ(J,K)
            RETURN
            END

      C
      C  PROCEDURA PENTRU DETERMINAREA VALORII MAXIME SI A POZITIEI
      C  ACESTEI VALORI INTR-UN STR ORDONAT DE NUMERE REALE
            SUBROUTINE MAXIM
            COMMON M,N/MAX/D(20),DMAX,JMAX
            DMAX=D(1)
            JMAX=1
            DO 2 I=2,N
            IF(DMAX.GE.D(I)) GO TO 1
            DMAX=D(I)
            JMAX=I
            GO TO 2
          1 DMAX=DMAX
            JMAX=JMAX
          2 CONTINUE
            RETURN
            END

      C
      C  PROCEDURA PENTRU DETERMINAREA VALORII MINIME SI A POZITIEI
      C  ACESTEI VALORI INTR-UN STR ORDONAT DE NUMERE REALE
            SUBROUTINE MINIM
            COMMON M,N/MIN/TETA(20),TETA0,IMIN
            TETA0=TETA(1)
            IMIN=1
            DO 2 I=2,M
            IF(TETA0.LT.TETA(I)) GO TO 1
            TETA0=TETA(I)
            IMIN=I
            GO TO 2
          1 TETA0=TETA0
            IMIN=IMIN
          2 CONTINUE
            RETURN
            END

      C
      C  PROCEDURA PENTRU DETERMINAREA NUMARULUI COMBINARILOR DE NC LUATE CITE M
            SUBROUTINE COMBIN(M,NC,NS)
            K=M-1
            JP=NC
            DO 1 I=1,K
          1 JP=JP*(NC-I)
            KF=1
            DO 2 I=2,M
          2 KF=KF*I
            NS=JP/KF
            RETURN
            END
```

Lista 6.1

Scanned with OKEN Scanner

```
 RUN
 STARTED    DATELE PROBLEMEI SINT URMATOARELE:

 MATRICEA COEFICIENTILOR DIN SISTEMUL DE RESTRICTII COMPLETAT CU VARIABILELE DE ECART
          (PE ULTIMA COLOANA SINT TERMENII LIBERI)
  1.000      1.000      1.000      1.000      1.000       .000       .000     15.000
  7.000      5.000      3.000      2.000       .000      1.000       .000      1.000
  3.000      5.000     10.000     15.000       .000       .000      1.000      1.000

 VALORILE COEFICIENTILOR DIN FUNCTIA OBIECTIV (IN ORDINEA NATURALA A INDICILOR):
   60.000     60.000     90.000     90.000       .000       .000       .000

 VALORILE ALESE INITIAL PENTRU ELEMENTELE SOLUTIEI DE BAZA (IN ORDINEA NATURALA A INDICILOR):
    .0000      .0000      .0000      .0000    15.0000     1.0000     1.0000

 COEFICIENTII COLOANELOR DIN MATRICEA A INTRODUSE INITIAL IN BAZA:
5    6    7

 COEFICIENTII COLOANELOR DIN MATRICEA A CARE INITIAL NU INTRA IN BAZA:
1    2    3    4


        REZULTATELE FINALE, DUPA  2 ITERATII, SINT URMATOARELE:
        ********************************************************

     VALOAREA MAXIMA A FUNCTIEI OBIECTIV:     19.79


        SOLUTIA OPTIMALA
        ----------------


        X( 1)=   .1148
        X( 2)=   .0000
        X( 3)=   .0656
        X( 4)=   .0000
        X( 5)= 14.8197
        X( 6)=   .0000
        X( 7)=   .0000
```

Lista 6.2

Scanned with OKEN Scanner

```
 RUN
 STARTED    DATELE PROBLEMEI SINT URMATOARELE:

 MATRICEA COEFICIENTILOR DIN SISTEMUL DE RESTRICTII COMPLETAT CU VARIABILELE DE ECART
           (PE ULTIMA COLOANA SINT TERMENII LIBERI)
  3.000       5.000       1.000        .000       15.000
  6.000       2.000        .000       1.000       24.000

 VALORILE COEFICIENTILOR DIN FUNCTIA OBIECTIV (IN ORDINEA NATURALA A INDICILOR):
  200.000     100.000         .000         .000

 VALORILE ALESE INITIAL PENTRU ELEMENTELE SOLUTIEI DE BAZA (IN ORDINEA NATURALA A INDICILOR):
    .0000       .0000      15.0000      24.0000

 COEFICIENTII COLOANELOR DIN MATRICEA A INTRODUSE INITIAL IN BAZA:
3    4

 COEFICIENTII COLOANELOR DIN MATRICEA A CARE INITIAL NU INTRA IN BAZA:
1    2


              REZULTATELE FINALE, DUPA  2 ITERATII, SINT URMATOARELE:
              ******************************************************

         VALOAREA MAXIMA A FUNCTIEI OBIECTIV:     825.00


              SOLUTIA OPTIMALA
              ----------------


              X( 1)=  3.7500
              X( 2)=   .7500
              X( 3)=   .0000
              X( 4)=   .0000


*STOP*
```


Lista 6.3

Scanned with OKEN Scanner

```
DATELE PROBLEMEI SINT URMATOARELE:

MATRICEA COEFICIENTILOR DIN SISTEMUL DE RESTRICTII COMPLETAT CU VARIABILELE DE ECART
          (PE ULTIMA COLOANA SINT TERMENII LIBERI)
2,000     2,500     3,000     3,500     4,200     1,000     ,000      ,000      ,000      ,000      ,000      300,000
1,000      ,000      ,000      ,000      ,000      ,000     1,000      ,000      ,000      ,000      ,000       70,000
 ,000     1,000      ,000      ,000      ,000      ,000      ,000     1,000      ,000      ,000      ,000       30,000
 ,000      ,000     1,000      ,000      ,000      ,000      ,000      ,000     1,000      ,000      ,000       40,000
 ,000      ,000      ,000     1,000      ,000      ,000      ,000      ,000      ,000     1,000      ,000       30,000
 ,000      ,000      ,000      ,000     1,000      ,000      ,000      ,000      ,000      ,000     1,000       25,000

VALORILE COEFICIENTILOR DIN FUNCTIA OBIECTIV (IN ORDINEA NATURALA A INDICILOR):
   5,000     6,000     6,500     7,000     8,000      ,000      ,000      ,000      ,000      ,000
    ,000

VALORILE ALESE INITIAL PENTRU ELEMENTELE SOLUTIEI DE BAZA (IN ORDINEA NATURALA A INDICILOR):
    ,0000     ,0000     ,0000     ,0000     ,0000   300,0000    70,0000   50,0000   40,0000   30,0000
25,0000

COEFICIENTII COLOANELOR DIN MATRICEA A INTRODUSE INITIAL IN BAZA:
6    7    8    9   10   11

COEFICIENTII COLOANELOR DIN MATRICEA A CARE INITIAL NU INTRA IN BAZA:
1    2    3    4    5

          REZULTATELE FINALE, DUPA  6 ITERATII, SINT URMATOARELE:
          **********************************************************

VALOAREA MAXIMA A FUNCTIEI OBIECTIV:     716,67

          SOLUTIA OPTIMALA
          ----------------

          X( 1)=  70,000
          X( 2)=  50,000
          X( 3)=    ,000
          X( 4)=    ,000
          X( 5)=   8,333
          X( 6)=    ,000
          X( 7)=    ,000
          X( 8)=    ,000
          X( 9)=  40,000
          X(10)=  30,000
          X(11)=  16,667
```

Lista 6.4

1° $max\ F = min\ G$.

2° Cantităţile de forma $(F_j - c_j) \geqslant 0$, utilizate la aplicarea metodei simplex, devin în cazul programelor duale:

$F_k - c_k$ pentru programul (6—48),

$-(G_k - b_k)$ pentru programul (6—49).

Pentru a înlesni scrierea şi explicaţiile, vom utiliza următorul mod de numerotare al variabilelor: cu indici de la 1 la n pentru variabilele efective ale programului (6—48), care vor fi $x_1, x_2, \ldots, x_n$; cu indici de la $n+1$ la $n+m$ pentru variabilele de ecart din (6—48), adică $x_{n+1}, x_{n+2}, \ldots, x_{n+m}$; cu indici de la $n+1$ la $n+m$ pentru variabilele efective ale programului (6—49), care vor fi $y_{n+1}, y_{n+2}, \ldots, y_{n+m}$ şi cu indici de la 1 la n pentru variabilele de ecart din (6—49), adică $y_1, y_2, \ldots, y_n$.

Diferenţele de mai sus pot fi utilizate pentru obţinerea soluţiilor în felul următor: $F_k - c_k$ pentru soluţiile programului (6—49) şi $G_k - b_k$ pentru soluţiile celuilalt program (dual), adică (6—48), ceea ce — cu notaţiile de mai sus — se poate exprima prin:

$$\begin{aligned} &F_k - c_k = y_k, \quad y_k \geqslant 0, \quad k = n+1, \quad n+2, \ldots, n+m; \\ &b_k - G_k = x_k, \quad x_k \geqslant 0, \quad k = 1, 2, \ldots, n. \end{aligned} \tag{6—50}$$

3° Oricărei probleme de programare liniară (să-i spunem *program primar*) i se poate ataşa o altă problemă de programare liniară, duală, exprimată de un *program dual*. Între soluţiile optimale ale celor două programe (primar şi dual) existînd legătura dată de expresiile (6—50), rezultă că rezolvînd o problemă de programare liniară am determinat, implicit, şi soluţia problemei duale.

De multe ori este mai simplu (ca volum de calcul şi memorie operativă) să rezolvăm problema duală a problemei date şi, apoi, prin relaţiile (6—50) să determinăm soluţia optimală a problemei primare.

Ex. 6—9. Să se scrie programul dual al problemei din *Ex. 6—7* şi să se determine soluţia sa optimală.

Programul primar (dat în *Ex. 6—7*) este următorul:

$$\begin{cases} F = 60\ x_1 + 60x_2 + 90x_3 + 90x_4 = max \\ x_1 + x_2 + x_3 + x_4 \leqslant 15 \\ 7x_1 + 5x_2 + 3x_3 + 2x_4 \leqslant 1 \\ 3x_1 + 5x_2 + 10x_3 + 15x_4 \leqslant 1 \end{cases}$$

şi are forma (6—48). Atunci, programul dual de forma (6—49), rezultă imediat din tabela:

	x_1	x_2	x_3	x_4	b_i	$i = 1, 3, 2,$
y_1	1	1	1	1	15	
y_2	7	5	3	2	1	
y_3	3	5	10	15	1	
c_j	60	60	90	90		$j = 1, 2, 3, 4$

Scanned with OKEN Scanner

pe baza căreia se poate scrie direct programul dual:

$$\begin{cases} G = 15\ y_1 + y_2 + y_3 = min \\ y_1 + 7y_2 + 3y_3 \geqslant 60 \\ y_1 + 5y_2 + 5y_3 \geqslant 60 \\ y_1 + 3y_2 + 10y_3 \geqslant 90 \\ y_1 + 2y_2 + 15y_3 \geqslant 90 \end{cases}$$

După cum se vede, programul primar are $m = 3$ restricţii şi $n = 4$ variabile efective x_i, iar programul dual are 4 restricţii şi 3 variabile efective y_1, y_2, şi y_3.

În ceea ce priveşte soluţia finală a programului dual, deoarece $max\ F = 780/61$ (v. *Ex. 6—7*), atunci (conform proprietăţii 2° de mai sus) rezultă că $min\ G = 780/61$, pentru care soluţia optimală se obţine aplicînd egalităţile (6—50) şi utilizînd tabela 6—5.

Tabela 6—5

variabile duale	$-y_4$	$-y_5$	$-y_6$	$-y_7$	y_1	y_2	y_3
variabile primare	x_1	x_2	x_3	x_4	x_5	x_6	x_7
soluţia optimală primară	7/61	0	4/61	0	904/61	0	0
$F_k - c_k$	0	240/61	0	1920/61	0	330/61	450/61

adică:

$$y_1 = F_{n+1} - c_{n+1} \quad F_5 - c_5 = 0,$$
$$y_2 = F_{n+2} - c_{n+2} = F_6 - c_6 = 330/61 \quad \text{şi}$$
$$y_3 = F_{n+3} - c_{n+3} = F_7 - c_7 = 450/61.$$

Metode de rezolvare a problemelor de transport

După cum am mai arătat (v. *Ex. 6—4* şi expresiile 6—18, 6—19) modelul unei probleme de transport este format din $m + n - 1$ ecuaţii liniare cu $m \cdot n$ necunoscute notate cu x_{ij} ($i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$). Deşi modelele (6—19/1) şi (6—19/2) sînt formate din $m + n$ ecuaţii, numai $m + n - 1$ sînt distincte deoarece o ecuaţie (din cele $m + n$ ale modelului) se poate obţine din celelalte printr-o combinaţie liniară rezultată din faptul că $a_1 + a_2 + \ldots + a_m = b_1 + b_2 + \ldots + b_n$.

Prin urmare, trebuie rezolvat un sistem de $m + n - 1$ ecuaţii algebrice liniare cu $m \cdot n$ necunoscute. Deoarece numărul ecuaţiilor este mai mic decît numărul necunoscutelor, problema poate avea mai multe soluţii; dintre aceste soluţii trebuie aleasă numai aceea care face ca funcţia obiectiv să aibă valoarea minimă (soluţia optimală).

Scanned with OKEN Scanner

Rezolvarea problemelor de transport se face, în majoritatea cazurilor, prin mai multe etape. În general, prima etapă constă în determinarea unei soluții inițiale; toate etapele următoare (cu excepția ultimei) au două faze: 1° verificarea soluției determinate în etapa precedentă pentru a se constata dacă aceasta este sau nu soluția optimală (dacă da, calculul ia sfîrșit și nu se mai execută cea de a doua fază); 2° îmbunătățirea soluției, în vederea executării unei noi etape (atunci cînd soluția s-a dovedit a nu fi cea optimală).

Determinarea soluției inițiale. Orice soluție a problemei transporturilor, care constă din $m + n - 1$ valori strict pozitive corespunzătoare variabilelor x_{ij}, se numește *soluție de bază nedegenerată*. Alegerea inițială a unei astfel de soluții se poate realiza prin numeroase procedee; în continuare vom prezenta două din cele mai uzuale.

1° *Procedeul colțului nord-vest.* Este denumit așa deoarece utilizează o reprezentare intuitivă a modelului sub forma unor tabele succesive în care întotdeauna se atribuie valori variabilei x_{ij} situată în colțul din „stînga-sus" (sau „nord-vest") al tabelei. Deoarece ultima valoare ce se atribuie este pentru necunoscuta x_{mn} situată în colțul din „dreapta-jos" (sau „sud-est") al tabelei, procedeul este denumit și procedeul „drumului nord-vest, sud-est".

Astfel, un model de forma (6—19) și (6—19/2) al unei probleme de transport se poate reprezenta și printr-o tabelă de felul tabelei 6—6.

Tabela 6—6

Depozite de aprovizionare ↓ / Stații de desfacere →	1	2	...	n	Cantitatea disponibilă în depozite a_i
1	x_{11}	x_{12}	...	x_{1n}	a_1
2	x_{21}	x_{22}	...	x_{2n}	a_2
.	.	.		.	.
.	.	.		.	.
.	.	.		.	.
m	x_{m1}	x_{m2}	...	x_{mn}	a_m
Cantitatea necesară stației b_j	b_1	b_2	...	b_n	TOTAL: $\sum_1^m a_i = \sum_1^n b_j$

Atunci, procedeul colțului nord-vest de atribuire a soluției inițiale constă în a da o valoare variabilei x_{11} (situată în colțul nord-vest al tabelei 6—5). Această valoare este valoarea minimă dintre a_1 și b_1, adică:

$$x_{11}^{(0)} = min\,\{a_1, b_1\},$$

unde notația (0) indică faptul că este vorba de o valoare inițială.

Astfel, dacă $b_1 < a_1$ atunci $x_{11}^{(0)} = b_1$ și conform ecuației (6—19/2) $x_{21}^{(0)} = x_{31}^{(0)} = \ldots = x_{m1}^{(0)} = 0$; dacă $a_1 < b_1$ atunci $x_{11}^{(0)} = a_1$ și așa cum

Scanned with OKEN Scanner

rezultă din (6—19/1) $x_{12}^{(0)} = x_{13}^{(0)} = \ldots = x_{1n}^{(0)} = 0$. În felul acesta, prima coloană (sau prima linie) a variabilelor din tabela 6—6 primește valori inițiale.

În continuare se întocmește o nouă tabelă de forma 6—5; astfel, să zicem că $b_1 < a_1$ și atunci noua tabelă (prezentată mai simplificat decît în 6—6) este:

	1	2	3	...	n	
1	$x_{11}^{(0)} = b_1$	x_{12}	x_{13}	...	x_{1n}	$a_1 - b_1$
2	0	x_{22}	x_{23}	...	x_{2n}	a_2
3	0	x_{32}	x_{33}	...	x_{3n}	a_3
.	.	.	.	...	.	.
.	.	.	.	...	.	.
.	.	.	.	...	.	.
m	0	x_{m2}	x_{m3}	...	x_{mn}	a_m
	$b_1 - b_1 = 0$	b_2	b_3	...	b_n	

Se consideră, și aici, tot variabila din colțul de nord-vest, adică x_{12}, căreia i se atribuie (ca valoare inițială) cea mai mică valoare dintre b_2 și $(a_1 - b_1)$. Se înțelege că s-a luat $a_1 - b_1$, deoarece cantitatea pe care o poate livra acum depozitul 1 este mai mică cu b_1 (prin cantitatea livrată stației de desfacere 1, deoarece $x_{11}^{(0)} = b_1$). Deci se va considera:

$$x_{12}^{(0)} = min\{(a_1 - b_1),\ b_2\} \text{ dacă } b_1 < a_1,$$

sau

$$x_{21}^{(0)} = min\{a_2,\ (b_1 - a_1)\} \text{ dacă } a_1 < b_1.$$

De exemplu, dacă $b_1 < a_1$ și $(a_1 - b_1) < b_2$ atunci soluția inițială se completează și cu valorile: $x_{12}^{(0)} = a_1 - b_1$, $x_{13}^{(0)} = x_{14}^{(0)} = \ldots = x_{1n}^{(0)} = 0$.

Operația continuă cu întocmirea unei noi tabele, în care colțul de nord-vest va fi ocupat de necunoscuta x_{22} căreia i se va atribui valoarea cea mai mică dintre a_2 și $b_2 - (a_1 - b_1)$ etc.

Evident că operația de atribuire a valorilor inițiale prin procedeul colțului nord-vest poate fi realizată și fără suportul grafic al tabelelor. Noi am utilizat și aceste tabele pentru a face o prezentare mai clară a procedeului.

Să observăm un fapt evident, acela că procedeul descris mai sus poate fi aplicat utilizînd și alt colț al tabelei cu variabile, ca de exemplu colțul nord-est. În acest caz prima necunoscută căreia i se atribuie valoarea inițială este x_{1n}, care devine $x_{1n}^{(0)} = min\{a_1,\ b_n\}$, iar ultima este necunoscuta x_{m1}, din colțul sud-vest (aceasta ar fi varianta „drumului nord-est, sud-vest" al procedeului, prezentat mai înainte prin cealaltă variantă a sa: „drumul nord-vest, sud-est").

S-ar putea proceda și în modul următor (dacă există motive de a minimiza calculul în cea de a doua etapă, de determinare a soluției optimale, prin reducerea numărului de iterații):

— se determină soluția inițială (adică tabloul $\mathbf{X}_{N-V}^{(0)}$) prin varianta colțului nord-vest și se calculează valoarea pe care o ia funcția obiectiv cu aceste valori $F_{N-V}^{(0)}$;

Scanned with OKEN Scanner

— se utilizează apoi și varianta colțului nord-est, determinîndu-se $\mathbf{X}^{(0)}_{N-E}$ și $F^{(0)}_{N-E}$;

— se alege, în final, ca soluție de bază inițială $\mathbf{X}^{(0)}$ una din cele două matrice determinate anterior, $\mathbf{X}^{(0)}_{N-V}$ sau $\mathbf{X}^{(0)}_{N-E}$, și anume aceea pentru care $F^{(0)} \in \{F^{(0)}_{N-V}, F^{(0)}_{N-E}\}$ este mai mică.

Ex. 6—10. Vom exemplifica procedeul colțului nord-vest de alegere a soluțiilor inițiale la problema de transport prezentată în *Ex. 6—4* (v. tabela 6—4 și relațiile 6—20, 6—21). Aplicînd direct, prin tabele, cele arătate mai înainte obținem în mod succesiv:

D_i \ S_j	1	2	3	4	5	a_i
1	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}	16
2	x_{21}	x_{22}	x_{23}	x_{24}	x_{25}	8
3	x_{31}	x_{32}	x_{33}	x_{34}	x_{35}	6
b_j	8	5	7	4	6	TOTAL = 30

— În cazul variantei „drumul nord-vest, sud-est":

1) deoarece $b_1 < a_1$ $(8 < 16)$ atunci $x^{(0)}_{11} = 8$, $x^{(0)}_{21} = 0$ și $x^{(0)}_{31} = 0$, rezultînd tabela:

	1	2	3	4	5	
1	8	x_{12}	x_{13}	x_{14}	x_{15}	$16 - 8 = 8$
2	0	x_{22}	x_{23}	x_{24}	x_{25}	8
3	0	x_{32}	x_{33}	x_{34}	x_{35}	6
	$8-8=0$	5	7	4	6	

2) deoarece $b_2 < (a_1 - b_1)$, adică $5 < (16-8)$, atunci $x^{(0)}_{12} = 5$, $x^{(0)}_{22} = 0$ și $x^{(0)}_{32} = 0$, rezultînd o nouă tabelă:

	1	2	3	4	5	
1	8	5	x_{13}	x_{14}	x_{15}	$16 - 8 - 5 = 3$
2	0	0	x_{23}	x_{24}	x_{25}	8
3	0	0	x_{33}	x_{34}	x_{35}	6
	0	$5-5=0$	7	4	6	

3) deoarece $(a_1 - b_1 - b_2) < b_3$ $(3 < 7)$ atunci $x^{(0)}_{13} = 3$, $x^{(0)}_{14} = 0$ și $x^{(0)}_{15} = 0$, rezultînd o nouă tabelă:

	1	2	3	4	5	
1	8	5	3	0	0	$16 - 8 - 5 - 3 = 0$
2	0	0	x_{23}	x_{24}	x_{25}	8
3	0	0	x_{33}	x_{34}	x_{35}	6
	0	0	$7-3=4$	4	6	

Scanned with OKEN Scanner

4) deoarece $b_3^{(3)} < a_2$ $(4 < 8)$ atunci $x_{23}^{(0)} = 4$ şi $x_{33}^{(0)} = 0$, după care tabelul ia forma :

1	2	3	4	5	
8	5	3	0	0	0
0	0	4	x_{24}	x_{25}	$8 - 4 = 4$
0	0	0	x_{34}	x_{35}	6
0	0	0	4	6	

5) deoarece $b_4 = a_2^{(4)} = 4$ atunci $x_{24}^{(0)} = 4$, $x_{34}^{(0)} = 0$ şi $x_{25}^{(0)} = 0$ rezultînd în final $x_{35}^{(0)} = 6$ şi deci următoarele valori iniţiale pentru necunoscutele x_{ij} $(i = 1, 2, 3,\ j = 1, 2, \ldots, 5)$:

$X_{N-V}^{(0)} =$

	1	2	3	4	5	
1	8	5	3	0	0	= 16
2	0	0	4	4	0	= 8
3	0	0	0	0	6	= 6
	= 8	=5	=7	=4	=6	

Soluţia $X_{N-V}^{(0)}$ de mai sus este o soluţie degenerată deoarece are numai şase valori diferite de zero (deci mai puţin decît $m + n - 1 = 7$).

— În cazul variantei „drumului nord-est, sud-vest" :

1) deoarece $b_5 < a_1$ $(6 < 16)$ atunci $x_{15}^{(0} = 6$, $x_{26}^{(0)} = 0$ şi $x_{35}^{(0)} = 0$, rezultînd tabela :

	1	2	3	4	5	
1	x_{11}	x_{12}	x_{13}	x_{14}	6	$16 - 6 = 10$
2	x_{21}	x_{22}	x_{23}	x_{24}	0	8
3	x_{31}	x_{32}	x_{33}	x_{34}	0	6
	8	5	7	4	$6-6=0$	

2) deoarece $b_5 < (a_1 - b_6)$, adică $4 < (16 - 6)$, se ia : $x_{14)}^{(0)} = 4$, $x_{24}^{(0)} = 0$ şi $x_{34}^{(0)} = 0$, rezultînd o nouă tabelă :

	1	2	3	4	5	
1	x_{11}	x_{12}	x_{13}	4	6	$16 - 6 - 4 = 6$
2	x_{21}	x_{22}	x_{23}	0	0	8
3	x_{31}	x_{32}	x_{33}	0	0	6
	8	5	7	$4-4=0$	0	

Scanned with OKEN Scanner

3) deoarece $a_1^{(2)} < b_3$, adică $(16-6-4) < 7$, atunci considerăm $x_{13}^{(0)} = a_1^{(2)} = 6$, de unde rezultă $x_{12}^{(0)} = x_{11}^{(0)} = 0$ şi o nouă tabelă:

	1	2	3	4	5	
1	0	0	6	4	6	0
2	x_{21}	x_{22}	x_{23}	0	0	8
3	x_{31}	x_{32}	x_{33}	0	0	6
	8	5	$7-6=1$	0	0	

4) pentru că $b_3^{(3)} < a_2$ $(1 < 8)$ atunci $x_{23}^{(0)} = 1$ şi $x_{33}^{(0)} = 0$, rezultînd o altă tabelă:

	1	2	3	4	5	
1	0	0	6	4	6	0
2	x_{21}	x_{22}	1	0	0	7
3	x_{31}	x_{32}	0	0	0	6
	8	5	0	0	0	

5) deoarece $b_2 < a_2^{(4)}$ $(5 < 7)$ atunci $x_{22}^{(0)} = 5$ şi $x_3^{(0)} = 0$, rezultînd imediat $x_{21}^{(0)} = 7-5 = 2$ şi (pentru ultima variabilă, din colţul „sud-vest") $x_{31}^{(0)} = 8 - x_{21}^{(0)} - x_{11}^{(0)} = 8 - 2 - 0 = 6$.

Obţinem deci următoarele valori iniţiale $\mathbf{X}_{N-E}^{(0)}$ (adică matricea soluţiei iniţiale determinată prin varianta „colţul nord-est"):

$$\mathbf{X}_{N-E}^{(0)} = \begin{array}{|c|c|c|c|c|l} \hline 0 & 0 & 6 & 4 & 6 & = 16 \\ \hline 2 & 5 & 1 & 0 & 0 & = 8 \\ \hline 6 & 0 & 0 & 0 & 0 & = 6 \\ \hline = 8 & = 5 & = 7 & = 4 & = 6 & \end{array}$$

2° *Procedeul Vogel.* Se pleacă de la datele problemei aranjate într-o tabelă de forma *6—4* (v. *Ex.* 6—4), în care se înscriu cheltuielile de transport c_{ij}. Această tabelă se completează cu o coloană şi o linie suplimentară, denumite coloană/linie cu diferenţe, care se întocmesc astfel: se alege costul minim din coloana/linia considerată şi se scade din costul imediat următor, obţinîndu-se nişte diferenţe care se înscriu pe coloana şi linia suplimentară.

După stabilirea acestor diferenţe, se determină linia/coloana pentru care a rezultat cea mai mare diferenţă şi căsuţa (i, j) interioară a tabelei (de pe această linie/coloană) în care este costul cel mai mic; necunoscutei corespunzătoare acestei căsuţe i se atribuie cantitatea maximă ce se poate transporta (adică valoarea $min\,\{a_i, b_j\}$). După aceasta se recalculează

Scanned with OKEN Scanner

coloana şi linia cu diferenţe, fără să se mai ţină seama de coloana/linia cu costuri corespunzătoare depozitului de livrare sau staţiei de primire care şi-au lichidat disponibilităţile sau şi-au asigurat necesarul de produse. Operaţiile se realizează iterativ, pînă se epuizează toate disponibilităţile de livrare şi se asigură necesarul la toate staţiile, aşa cum se arată numeric în exemplul prezentat în continuare.

Ex. 6—11. Să se determine soluţia iniţială prin procedeul Vogel în cazul problemei din *Ex. 6—4* (v. tabela 6—4), care a făcut şi obiectul exemplului 6—10.

Vom utiliza tabelele cu costul transportului, în interiorul cărora vom scrie, subliniat, şi valorile atribuite iniţial necunoscutelor x_{ij}. Tabela iniţială (aşa cum rezultă din tabela 6—4), completată cu linia şi coloana cu diferenţe este următoarea:

costuri c_{ij}

D_i \ S_j	1	2	3	4	5	Coloana cu diferenţe	a_i
1	3	5	2	3	2	1 = 3 − 2	16
2	1	3	4	8	5	2 = 3 − 1	8
3	2	8	1	2	3	1 = 2 − 1	6
Linia cu diferenţe	1=2−1	2=5−3	1=2−1	1=3−2	1=3−2	—	—
b_j	8	5	7	4	6	—	TOTAL: 30

1) Cea mai mare diferenţă este 2 şi să o considerăm pe cea de pe linia cu diferenţe în dreptul coloanei *2*; pe această coloană cel mai mic este $c_{22} = 3$, din „căsuţa" *(2, 2)*. În această căsuţă înscriem valoarea pe care o vom atribui iniţial lui x_{22} şi anume $x_{22}^{(0)} = min\{5, 8\} = 5$. Recalculăm coloana şi linia cu diferenţe (fără să ţinem seama de coloana *2)* şi obţinem o nouă tabelă:

c_{ij} $\underline{x_{ij}^{(0)}}$

P_i \ S_j	1	2	3	4	5	Coloana cu diferenţe	$a_i^{(1)}$
1	3	5 $\underline{0}$	2	3	2	1 = 3 − 2	16
2	1	3 $\underline{5}$	4	8	5	3 = 4 − 1	3=8−5
3	2	8 $\underline{0}$	1	2	3	1 = 2 − 1	6
Linia cu diferenţe	1=2−1	×	1=2−1	1=3−2	1=3−2	—	—
$b_j^{(1)}$	8	0=5−5	7	4	6	—	TOTAL: 30

Scanned with OKEN Scanner

2) Cea mai mare diferență este 3, de pe coloana cu diferențe în dreptul liniei *2*; pe această linie costul cel mai mic este $c_{21} = 1$, din căsuța *(2, 1)*. În această căsuță înscriem pe $x_{21}^{(0)} = min\{8, 3\} = 3$. Recalculăm coloana și linia cu diferențe (fără să mai ținem seama de linia *2*) și obținem o nouă tabelă:

c_{ij} $\underline{x}_{ij}^{(0)}$

S_j / D_i	*1*	*2*	*3*	*4*	*5*	Coloana cu diferențe	$a_i^{(2)}$
1	3	5 0	2	3	2	1 = 3 − 2	16
2	1 3	3 5	4 0	8 0	5 0	×	0
3	2	8 0	1	2	3	1 = 2 − 1	6
Linia cu diferențe	1=3−2	×	1=2−1	1=3−2	1=3−2	—	—
$b_j^{(2)}$	5=8−3	0	7	4	6	—	TOTAL: 30

3) Deoarece toate diferențele sînt egale (!) putem să facem orice opțiune. De pildă, considerăm coloana 1 și în „căsuța" *(3, 1)*, pentru care avem cel mai mic cost rămas pe coloana *1*, înscriem $x_{31}^{(0)} = min\{5, 6\} = 5$. Deoarece depozitul D_3 mai poate livra numai o unitate (din disponibilul de 6, au fost livrate 5 unități stației S_1), atunci considerăm $x_{33}^{(0)} = 1$ deoarece pe linia *3* costul cel mai mic este $c_{33} = 1$. De aici rezultă și alte valori inițiale: $x_{11}^{(0)} = x_{34}^{(0)} = x_{35}^{(0)} = 0$.

În sfîrșit, cantitatea $a_1 = 16$ trebuie repartizată stațiilor S_3, S_4 și S_5, adică: $x_{13}^{(0)} = b_3 - x_{33}^{(0)} = 7 - 1 = 6$, $x_{14}^{(0)} = b_4 = 4$ și $x_{15}^{(0)} = b_5 = 6$.

Rezultă, deci, următorul plan inițial de transport:

$X_V^{(0)} =$

	1	*2*	*3*	*4*	*5*	
1	0	0	6	4	6	= 16
2	3	5	0	0	0	= 8
3	5	0	1	0	0	= 6
	= 8	= 5	= 7	= 4	= 6	

Comparînd, prin funcția obiectiv (6—20), soluțiile inițiale alese pentru aceeași problemă prin trei metode diferite, obținem:

— în cazul procedeului „colțul nord-vest" (v. *Ex. 6-10*):

$$F_{N-V}^{(0)} = 3 \cdot 8 + 5 \cdot 5 + 2 \cdot 3 + 4 \cdot 4 + 8 \cdot 4 + 3 \cdot 6 = 111;$$

— în cazul procedeului „colțul nord-est" (v. *Ex. 6 —10*):

$$F_{N-E}^{(0)} = 2 \cdot 6 + 3 \cdot 4 + 2 \cdot 6 + 1 \cdot 2 + 3 \cdot 5 + 4 \cdot 1 + 2 \cdot 6 = 69;$$

— în cazul metodei Vogel (v. *Ex. 6—11*):

$$F_V^{(0)} = 2 \cdot 6 + 3 \cdot 4 + 2 \cdot 6 + 1 \cdot 3 + 3 \cdot 5 + 1 \cdot 1 = 65.$$

Scanned with OKEN Scanner

Deci, prin metoda Vogel am obținut o soluție inițială „mai bună" ($F_V^{(0)} < F_{N-E}^{(0)} < F_{N-V}^{(0)}$), ceea ce va duce la mărirea vitezei de convergență în rezolvarea iterativă a problemei în etapa a doua de calcul (durată mai mică a calculului).

Mai remarcăm, comparînd soluțiile obținute prin cele două variante ale procedeului colțului, că în cazul problemei din *Ex. 6—10*, este mai bine să alegem ca soluție de bază inițială valorile stabilite prin varianta „drumului nord-est, sud-vest". Aceasta deoarece funcția obiectiv fiind mult mai mică pentru „valorile nord-est" ($F_{N-E}^{(0)} = 69 < F_{N-V}^{(0)} = 111$), în etapa de determinare a soluției optimale vom efectua mult mai puține iterații decît în cazul în care am fi ales ca soluție inițială tabloul $\mathbf{X}_{N-V}^{(0)}$ rezultat prin varianta „drumului nord-vest, sud-est" (v. exemplul 6—12).

Totuși, în ceea ce privește rezolvarea problemelor de transport prin utilizarea calculatoarelor automate, menționăm că în exclusivitatea lor, pachetele de programe specializate pentru problemele de transport folosesc ca procedeu pentru determinarea soluției de bază inițială procedeul „drumului nord-vest, sud-est" (ca, de exemplu, programul de transport TRANSPOR/IRIS-50, implementat și la sistemele FELIX C-256 [32]). S-a preferat alegerea procedeului colțului (cu oricare din cele două variante ale sale: „drumul nord-vest, sud-est" sau „drumul nord-est, sud-vest") datorită, pe de o parte, simplității algoritmului său (care asigură un program calculator foarte compact) și pe de altă parte, faptului că iterațiile etapei următoare (de optimizare a soluției) sînt ele însele foarte rapide și — ca urmare — chiar dacă rezultă un număr de iterații mai mare în această etapă, timpul-calculator cîștigat prin folosirea unei alte soluții inițiale ameliorată („mai bună" decît $\mathbf{X}_{N-V}^{(0)}$) nu este în general important.

Determinarea soluției optimale. După cum am mai arătat, soluția optimală se poate determina printr-un proces iterativ de corectare a unei soluții de bază, prima (soluția de bază inițială) putînd fi aleasă așa cum am văzut în paragraful precedent. Avînd o soluție de bază (inițială sau obținută după o anumită iterație), se verifică dacă ea este optimală sau nu, în acest de al doilea caz trecîndu-se la îmbunătățirea soluției analizate ș.a. m.d. Există numeroase metode pentru găsirea soluției optimale, una din cele mai simple fiind cunoscuta *metodă distributivă*; prezentăm în continuare o variantă a acestei metode (metoda distributivă modificată).

Fiind dată tabela problemei (de forma 6—4), fiecărei linii i i se asociază un număr u_i și fiecărei coloane j — un număr v_j astfel încît pentru fiecare căsuță (i, j), ocupată de costul transportului c_{ij}, pentru care din iterația precedentă $x_{ij} \neq 0$, să avem egalitatea:

$$u_i + v_j = c_{ij} \quad i \in \{1, 2, \ldots, m\}, \quad j \in \{1, 2, \ldots, n\}.$$

Aceste egalități formează un sistem de $n + m - 1$ ecuații cu $n + m$ necunoscute u_i și v_j, care se pot determina considerînd, de exemplu, $u_1 = 0$. Valorile u_i și v_j, astfel obținute, se scriu pe o coloană și pe o linie a unei tabele în interiorul căreia, în fiecare căsuță (i, j), se scriu valorile $u_i + v_j$, așa ca în tabela 6—7.

Scanned with OKEN Scanner

Tabela 6—7

u_i \ v_j	v_1	v_2	v_3	...	v_n
u_1	$u_1 + v_1$	$u_1 + v_2$	$u_1 + v_3$	...	$u_1 + v_n$
u_2	$u_2 + v_1$	$u_2 + v_2$	$u_2 + v_3$	...	$u_2 + v_n$
u_3	$u_3 + v_1$	$u_3 + v_2$	$u_3 + v_3$	...	$u_3 + v_n$
⋮	⋮	⋮	⋮	⋮ ⋮ ⋮	⋮
u_m	$u_m + v_1$	$u_m + v_2$	$u_m + v_3$	...	$u_m + v_n$

Apoi, elementele din interiorul tabelei 6—7 se scad din costurile tabelei problemei (de tipul tabelei 6—4) și se obține o nouă tabelă în interiorul căreia se găsesc, în fiecare căsuță (i, j) valorile $c_{ij} - (u_i + v_j)$ $i = 1, 2, \ldots, m$ și $j = 1, 2, \ldots, n$. Avînd în vedere cum au fost determinate numerele u_i și v_j, cel puțin $n + m - 1$ din valorile $c_{ij} - (u_i + v_j)$ vor fi nule, iar altele din aceste valori pot fi negative.

Urmează, în continuare, să se determine dacă soluția analizată este cea optimală. Criteriul de optimalitate constă în verificarea semnului valorilor $c_{ij} - (u_i + v_j)$ din ultima tabelă. Dacă nu există nici o astfel de valoare negativă atunci soluția cercetată este soluția optimală; dacă cel puțin o valoare $c_{ij} - (u_i + v_j)$ este negativă, atunci soluția analizată nu este cea optimală și — ca urmare — se va trece la îmbunătățirea acestei soluții.

Îmbunătățirea soluției se poate face prin „procedeul ciclului". Pornind din căsuța în care s-a găsit cel mai mare număr negativ în valoare absolută, se formează un ciclu, adică un contur poligonal cu unghiuri de 90° obținut prin trasarea unei linii închise ce trece numai prin căsuțe, în care $x_{ij} \neq 0$ și pleacă și se întoarce în căsuța cu valoare $c_{ij} - (u_i + v_j)$ negativă (pentru care $x_{ij} = 0$). Ciclul este, de obicei, un dreptunghi sau un pătrat ale cărui vîrfuri se numerotează cu *1*, *2*, *3* și *4*, începînd cu căsuța din care am plecat. Apoi, alegem cea mai mică cantitate x_{ij} din căsuțele corespunzătoare vîrfurilor cu numere de ordine pare; aceasta se adună la valorile x_{ij} din căsuțele tabelei cu soluții corespunzătoare vîrfurilor impare ale ciclului și se scade din valorile x_{ij} aflate în căsuțele corespunzătoare vîrfurilor pare. S-a obținut astfel o nouă soluție de bază, cu care se reia șirul operațiilor descrise mai sus, pînă cînd se ajunge la soluția optimală (v. *Ex. 6—12* și *Ex. 6—13*).

Ex. 6—12. Să se aplice metoda distributivă (modificată) pentru determinarea soluției optime a problemei din *Ex. 6—4*, alegînd ca soluție de bază inițială soluția determinată în *Ex. 6—10*, prin procedeul colțului nord-est.

Datele problemei, adică: costurile transportului c_{ij}, disponibilitățile de livrare a_i și necesarul la aprovizionări b_j $(i = 1, 2, 3, j = 1, 2, \ldots, 5)$, se pot aranja într-o tabelă de forma:

c_{ij}	1	2	3	4	5	$\leftarrow j$
1	3	5	2	3	2	16
2	1	3	4	8	5	8
3	2	8	1	2	3	6
$\uparrow i$	8	5	7	4	6	a_i / b_j

(în acest caz, au fost înscrise datele problemei din exemplul 6—4).

Pentru simplificarea scrierii, în cele ce vor urma vom folosi notațiile:

$$\mathbf{U} = \begin{bmatrix} u_1 & u_1 & \dots & u_1 \\ u_2 & u_2 & \dots & u_2 \\ \vdots & \vdots & & \vdots \\ u_m & u_m & \dots & u_m \end{bmatrix} \text{ cu } n \text{ coloane;}$$

$$\mathbf{V} = \begin{bmatrix} v_1 & v_2 & \dots & v_n \\ v_1 & v_2 & \dots & v_n \\ \vdots & \vdots & & \vdots \\ v_1 & v_2 & \dots & v_n \end{bmatrix} \text{ cu } m \text{ linii;}$$

$$\mathbf{COST} = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & c_{1n} \\ \vdots & \vdots & & \vdots \\ c_{m1} & c_{m2} & \dots & c_{mn} \end{bmatrix} = [c_{ij}];$$

$\mathbf{DIF} = \mathbf{COST} - (\mathbf{U} + \mathbf{V})$, care reprezintă tabloul cu valori

$$c_{ij} - (u_i + v_j) \quad (i = 1, 2, \dots, m, \; j = 1, 2, \dots, n);$$

și

$$\mathbf{X}^{(k)} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{21} \\ \vdots & \vdots & & \vdots \\ x_{m1} & x_{m2} & \dots & x_{mn} \end{bmatrix}^{(k)},$$ care reprezintă tabloul necunoscutelor după iterația k $(k = 0, 1, 2, \dots)$

$\mathbf{X}^{(0)}$ fiind soluția de bază inițială, determinată în prealabil.

Alegînd ca soluție de bază inițială tabloul $\mathbf{X}^{(0)}_{N-E}$ (v. *Ex. 6—10*), adică:

$$\mathbf{X}^{(0)}_{N-E} = \begin{bmatrix} 0 & 0 & 6 & 4 & 6 \\ 2 & 5 & 1 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 \end{bmatrix},$$

Scanned with OKEN Scanner

începem prima iterație din etapa de determinare a soluției optimale prin metoda distributivă modificată:

$\boxed{k = 1}$

— scriem, mai întîi, sistemul $u_i + v_j = c_{ij}$ pentru fiecare element din $\mathbf{X}^{(0)}$, pentru care $x_{ij} \neq 0$:

$$\begin{aligned} u_1 + v_3 &= 2 \\ u_1 + v_4 &= 3 \\ u_1 + v_5 &= 2 \\ u_2 + v_1 &= 1 \\ u_2 + v_2 &= 3 \\ u_2 + v_3 &= 4 \\ u_3 + v_1 &= 2; \end{aligned}$$

— rezolvăm sistemul de mai sus (de 7 ecuații cu 8 necunoscute u_i $i = 1, 2, 3$ și v_j $j = 1, 2, 3, 4, 5$) considerînd $u_1 = 0$.

Obținem: $u_2 = 2$, $u_3 = 3$, $v_1 = -1$, $v_2 = 1$, $v_3 = 2$, $v_4 = 3$ și $v_2 = 2$, adică:

$$\mathbf{U} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 \end{bmatrix} \text{ și } \mathbf{V} = \begin{bmatrix} -1 & 1 & 2 & 3 & 2 \\ -1 & 1 & 2 & 3 & 2 \\ -1 & 1 & 2 & 3 & 2 \end{bmatrix};$$

— determinăm sumele $u_i + v_j$, adică:

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} -1 & 1 & 2 & 3 & 2 \\ 1 & 3 & 4 & 5 & 4 \\ 2 & 4 & 5 & 6 & 5 \end{bmatrix};$$

— deoarece, conform datelor problemei din *Ex. 6—4*, matricea **COST** este:

$$\mathbf{COST} = \begin{bmatrix} 3 & 5 & 2 & 3 & 2 \\ 1 & 3 & 4 & 8 & 5 \\ 2 & 8 & 1 & 2 & 3 \end{bmatrix},$$

putem calcula matricea $\mathbf{DIF} = \mathbf{COST} - (\mathbf{U} + \mathbf{V})$, adică:

$$\mathbf{DIF} = \begin{bmatrix} 4 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 1 \\ 0 & 4 & -4 & -4 & -2 \end{bmatrix};$$

— deoarece matricea **DIF** are și elemente cu valori mai mici decît zero, adică deoarece există cel puțin o valoare $c_{ij} - (u_i + v_j) < 0$ $i \in \{1, 2, \ldots, m\}$ și $j \in \{1, 2, \ldots, n\}$, rezultă că soluția $\mathbf{X}^{(0)}$ nu este cea optimală și, deci, trecem la îmbunătățirea ei;

Scanned with OKEN Scanner

— îmbunătăţirea valorilor $\mathbf{X}^{(k-1)}$ o putem face prin procedeul ciclului. Astfel, deoarece elementul negativ cel mai mare în valoare absolută este în căsuţa (3, 3) din tabloul **DIF**, atunci ciclul în $\mathbf{X}^{(k-1)}$ se realizează pornind şi sfîrşind în căsuţa (3, 3) în care avem $x_{33}^{(0)} = 0$; acest ciclu se realizează astfel încît (plecînd şi întorcîndu-ne în căsuţa 3, 3) vîrfurile poligonului — cu excepţia primului — să se afle în căsuţe pentru care $x_{ij} \neq 0$, aşa cum este indicat în figura 6.3.

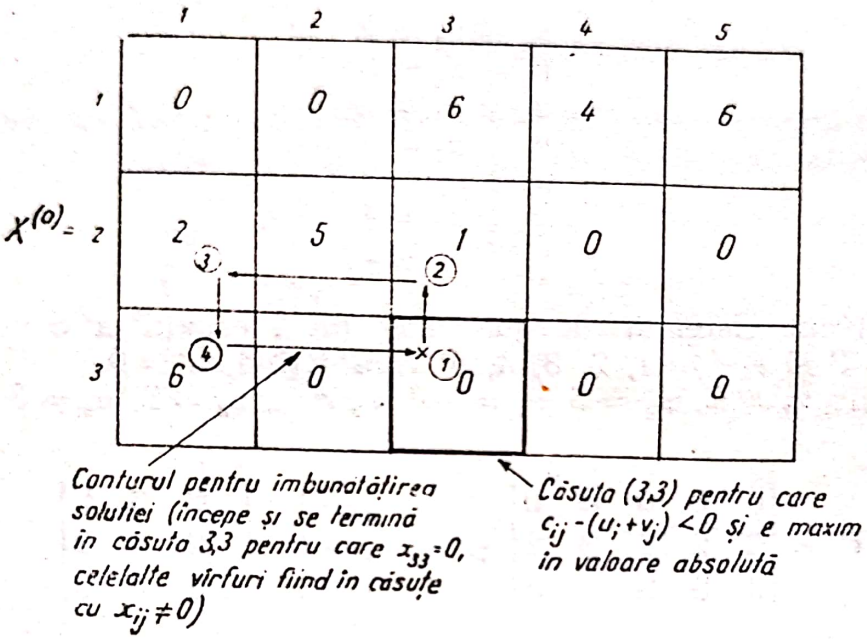


Fig. 6.3

După cum se vede în această figură, ciclul are colţurile pare în căsuţele unde sînt valorile: $x_{23}^{(0)} = 1$ şi $x_{31}^{(0)} = 6$ şi deoarece $\mu = min\{x_{23}^{(0)}, x_{31}^{(0)}\} = 1$, obţinem o nouă soluţie $\mathbf{X}^{(k)}$ (în cazul nostru $k = 1$), în care $x_{ij}^{(k)} = x_{ij}^{(k-1)}$, cu excepţia valorilor din căsuţele unde sînt colţurile pare ale conturului pentru care $x_{ij}^{(k)} = x_{ij}^{(k-1)} - \mu$ (în cazul nostru: $x_{23}^{(1)} = x_{23}^{(0)} - 1 = 1 - 1 = 0$ şi $x_{31}^{(1)} = x_{31}^{(0)} - 1 = 6 - 1 = 5$) şi a valorilor din căsuţele unde sînt colţurile impare ale conturului pentru care $x_{ij}^{(k)} = x_{ij}^{(k-1)} + \mu$ (în acest exemplu: $x_{21}^{(1)} = x_{21}^{(0)} + 1 = 2 + 1 = 3$ şi $x_{33}^{(1)} = x_{33}^{(0)} + 1 = 0 + 1 = 1$). Obţinem (în cazul acestui exemplu) următoarea nouă soluţie de la iteraţia $k = 1$:

$$\mathbf{X}^{(1)} = \begin{bmatrix} 0 & 0 & 6 & 4 & 6 \\ 3 & 5 & 0 & 0 & 0 \\ 5 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

Operaţiile se reiau, realizîndu-se o nouă iteraţie $k \to k + 1$ (în cazul acestui exemplu $k = 1 + 1 = 2$). Vom obţine:

$\boxed{k = 2}$

— sistemul de ecuaţii $u_i + v_j = c_{ij}$ (scrise pentru fiecare element din $\mathbf{X}^{(1)}$, pentru care $x_{ij} \neq 0$) este:

$$\begin{aligned} u_1 + v_3 &= 2 \\ u_1 + v_4 &= 3 \\ u_1 + v_5 &= 2 \\ u_2 + v_1 &= 1 \\ u_2 + v_2 &= 3 \\ u_3 + v_1 &= 2 \\ u_3 + v_3 &= 1 ; \end{aligned}$$

— dacă $u_1 = 0$, atunci: $u_2 = -2$, $u_3 = -1$, $v_1 = 3$, $v_2 = 5$, $v_3 = 2$, $v_4 = 3$ şi $v_5 = 2$, rezultînd:

$$\mathbf{U} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ -2 & -2 & -2 & -2 & -2 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix}, \quad \mathbf{V} = \begin{bmatrix} 3 & 5 & 2 & 3 & 2 \\ 3 & 5 & 2 & 3 & 2 \\ 3 & 5 & 2 & 3 & 2 \end{bmatrix},$$

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} 3 & 5 & 2 & 3 & 2 \\ 1 & 3 & 0 & 1 & 0 \\ 2 & 4 & 1 & 2 & 1 \end{bmatrix} \text{ şi } \mathbf{DIF} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 7 & 5 \\ 0 & 4 & 0 & 0 & 2 \end{bmatrix};$$

— deoarece tabloul **DIF** nu conţine nici un element negativ rezultă că $\mathbf{X}^{(k)}$ este soluţia optimală şi calculul ia sfîrşit aici.

Să observăm că în cazul acestui exemplu, soluţia optimală, care este $\mathbf{X}^{(1)}$, coincide cu soluţia determinată direct prin metoda lui Vogel (v. *Ex. 6—11*).

Să mai observăm, de asemenea, că în cazul acestei probleme nu puteam alege ca soluţie iniţială soluţia degenerată determinată prin metoda colţului nord-vest, $\mathbf{X}^{(0)}_{N-V}$ (v. *Ex. 6—10*), deoarece sistemul de ecuaţii $u_i + v_j = c_{ij}$ are numai 6 ecuaţii şi 8 necunoscute, ceea ce complică întrucîtva calculul de determinare a valorilor u_i şi v_j (v. *Ex. 6—13*).

Cazuri speciale. La rezolvarea problemelor de transport se pot ivi diferite situaţii speciale. Una din cele mai frecvente este aceea în care soluţia iniţială sau o soluţie intermediară conţine mai puţin decît $m+n-1$ valori strict pozitive (mai mari şi diferite de zero), adică este o soluţie de bază degenerată, sau nu conţine cicluri prin intermediul cărora să se poată îmbunătăţi soluţia.

Pentru a înţelege mai bine cum se procedează în astfel de situaţii speciale vom folosi un exemplu numeric.

Ex. 6—13. Să se rezolve problema de transport din *Ex. 6—4* utilizîndu-se ca soluţie iniţială soluţia determinată prin procedeul colţului nord-vest (v. *Ex. 6—10*).

În cazul problemei din exemplul 6—4, matricea costurilor este:

$$\mathbf{COST} = \begin{bmatrix} 3 & 5 & 2 & 3 & 2 \\ 1 & 3 & 4 & 8 & 5 \\ 2 & 8 & 1 & 2 & 3 \end{bmatrix},$$

Scanned with OKEN Scanner

iar soluţia iniţială, determinată la exemplul 6—10, este dată de tabloul:

$$\mathbf{X}^{(0)}_{N-V} = \begin{bmatrix} 8 & 5 & 3 & 0 & 0 \\ 0 & 0 & 4 & 4 & 0 \\ 0 & 0 & 0 & 0 & 6 \end{bmatrix},$$

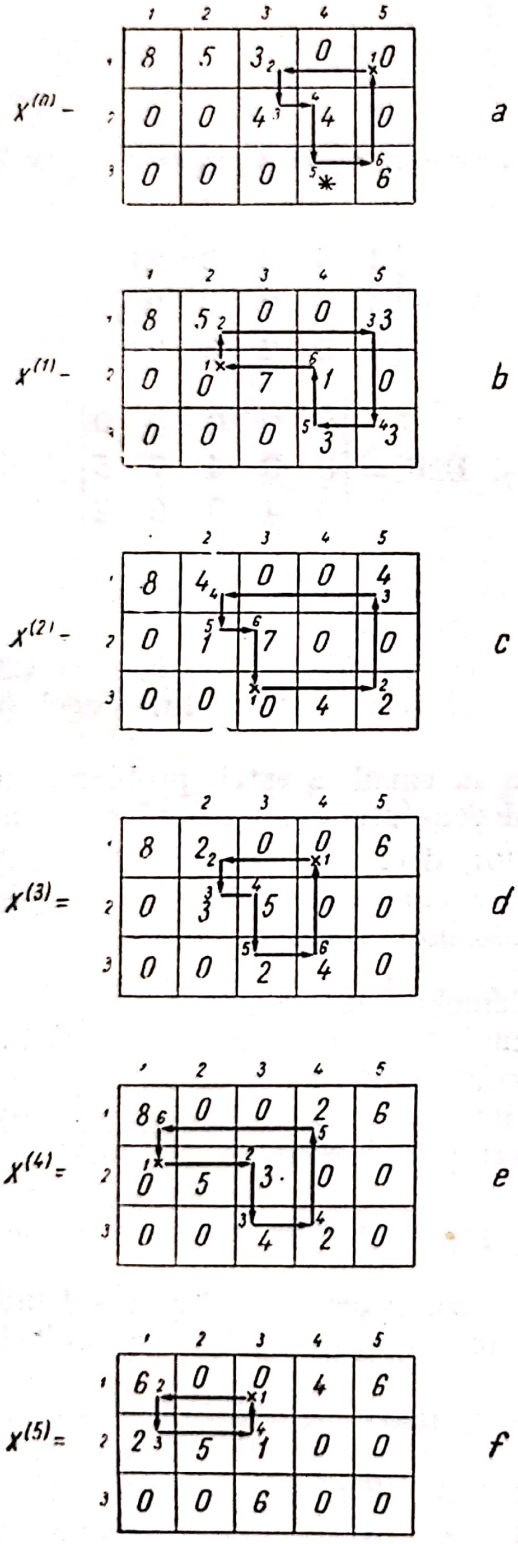


Fig. 6.4

care reprezintă o soluţie degenerată (deoarece are numai 6 valori mai mari decît zero, în timp ce $m + n - 1 = 7$, numărul de linii fiind $m = 3$ şi numărul de coloane $n = 5$).

Pentru a putea aplica metoda distributivă modificată, de determinare a soluţiei optimale, vom considera că unul din elementele tabloului $\mathbf{X}^{(0)}$ nu ar fi zero; să zicem că acesta ar fi x_{34}, din căsuţa (3, 4) a matricei $\mathbf{X}^{(0)}$ unde înscriem semnul * (v. fig. 6.4, *a*). În acest fel se consideră $x_{34} > 0$ şi se aplică întocmai metoda distributivă, aşa cum vom proceda şi noi în continuare, limitîndu-ne numai la comentariile strict necesare.

$\boxed{k = 1}$

Scriem sistemul $u_i + v_j = c_{ij}$ numai pentru $i \in \{1, 2, 3\}$ şi $j \in \{1, 2, 3, 4, 5\}$ pentru care în căsuţele matricei $\mathbf{X}^{(0)}$ avem $x_{ij} \neq 0$ (inclusiv x_{34} în a cărui căsuţă am înscris *):

$$\begin{aligned} u_1 + v_1 &= 3 \\ u_1 + v_2 &= 5 \\ u_1 + v_3 &= 2 \\ u_2 + v_3 &= 4 \\ u_2 + v_4 &= 8 \\ u_3 + v_4 &= 2 \\ u_3 + v_5 &= 3 \end{aligned}$$

care are soluţia

$$\begin{aligned} u_1 &= 0, & v_1 &= 3, \\ u_2 &= 2, & v_2 &= 5, \\ u_3 &= -4, & v_3 &= 2, \\ & & v_4 &= 6, \\ & & v_5 &= 7 \end{aligned}$$

rezultînd după aceasta:

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} 3 & 5 & 2 & 6 & 7 \\ 5 & 7 & 4 & 8 & 9 \\ -1 & 1 & -2 & 2 & 3 \end{bmatrix}$$

şi $\mathbf{DIF} = \begin{bmatrix} 0 & 0 & 0 & -3 & -5 \\ -4 & -4 & 0 & 0 & -4 \\ 3 & 7 & 3 & 0 & 0 \end{bmatrix}$

în care cel mai mic element (negativ) este în căsuța (*1*, *5*). De aceea ciclul pe tabloul $\mathbf{X}^{(0)}$ se realizează începînd din căsuța (*1*, *5*), trecînd numai prin căsuța pentru care $x_{ij} \neq 0$ și terminînd tot în căsuța (*1*, *5*); notăm, în ordinea parcursului (indicat cu săgeți), vîrfurile conturului obținut: *1*, *2*, *3*, *4*, *5* și *6*, așa cum este indicat în figura 6.4, *a*. Luînd numai valorile din căsuțele vîrfurilor cu număr de ordine par rezultă $\mu = min\ \{3, 4, 6\} = 3$ și următoarele noi valori pentru soluția îmbunătățită $\mathbf{X}^{(1)}$:

— pentru căsuțele cu vîrfuri pare: $x_{13}^{(1)} = x_{13}^{(0)} - \mu = 3 - 3 = 0$, $x_{24}^{(1)} = x_{24}^{(0)} - \mu = 4 - 3 = 1$ și $x_{35}^{(1)} = x_{35}^{(0)} - \mu = 6 - 3 = 3$;

— pentru căsuțele cu vîrfuri impare: $x_{15}^{(1)} = x_{15}^{(0)} + \mu = 0 + 3 = 3$, $x_{23}^{(1)} = x_{23}^{(0)} + \mu = 4 + 3 = 7$ și $x_{34}^{(1)} = x_{34}^{(0)} + \mu = 0 + 3$;

— pentru restul căsuțelor $x_{ij}^{(1)} = x_{ij}^{(0)}$.

Se obține astfel, următoarea nouă soluție (după prima iterație):

$$\mathbf{X}^{(1)} = \begin{bmatrix} 8 & 5 & 0 & 0 & 3 \\ 0 & 0 & 7 & 1 & 0 \\ 0 & 0 & 0 & 3 & 3 \end{bmatrix},$$

care nu mai este o soluție degenerată, deoarece are $m + n - 1 = 7$ valori mai mari decît zero.

Deoarece **DIF** are elemente negative, $\mathbf{X}^{(1)}$ nu este soluția optimală și calculul continuă cu o nouă iterație ($k = 2$).

$\boxed{k = 2}$

$$\begin{aligned} u_1 + v_1 &= 3 \\ u_1 + v_2 &= 5 \\ u_1 + v_5 &= 2 \\ u_2 + v_3 &= 4 \\ u_2 + v_4 &= 8 \\ u_3 + v_4 &= 2 \\ u_3 + v_5 &= 3 \end{aligned} \quad \rightarrow \quad \begin{aligned} u_1 &= 0 \\ u_2 &= 7 \\ u_3 &= 1 \end{aligned} \qquad \begin{aligned} v_1 &= 3 \\ v_2 &= 5 \\ v_3 &= 6 \\ v_4 &= 1 \\ v_5 &= 2 \end{aligned}$$

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} 3 & 5 & -3 & 1 & 2 \\ 10 & 12 & 4 & 8 & 9 \\ 4 & 6 & 2 & 2 & 3 \end{bmatrix}, \quad \mathbf{DIF} = \begin{bmatrix} 0 & 0 & 5 & 2 & 0 \\ -9 & -9 & 0 & 0 & -4 \\ -2 & 2 & -1 & 0 & 0 \end{bmatrix}.$$

Ciclul, pe $\mathbf{X}^{(1)}$ (fig. 6.4, *b*) se construiește pentru căsuța (*1*, *2*), rezultînd $\mu = min\ \{5, 1, 3\} = 1$ și o nouă soluție $x^{(2)}$ (v. fig. 6.4, *c*), care trebuie îmbunătățită deoarece **DIF** are valori negative.

Scanned with OKEN Scanner

$\boxed{k = 3}$

$$\begin{array}{l} u_1 + v_1 = 3 \\ u_1 + v_2 = 5 \\ u_1 + v_5 = 2 \\ u_2 + v_2 = 3 \\ u_2 + v_3 = 4 \\ u_3 + v_4 = 2 \\ u_3 + v_5 = 3 \end{array} \quad \rightarrow \quad \begin{array}{l} u_1 = 0 \\ u_2 = -2 \\ u_3 = 1 \end{array} \qquad \begin{array}{l} v_1 = 3 \\ v_2 = 5 \\ v_3 = 6 \\ v_4 = 1 \\ v_5 = 2 \end{array}$$

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} 3 & 5 & 6 & 1 & 2 \\ 1 & 3 & 4 & -1 & 0 \\ 4 & 6 & 7 & 2 & 3 \end{bmatrix}, \quad \text{DIF} = \begin{bmatrix} 0 & 0 & -4 & 2 & 0 \\ 0 & 0 & 0 & 9 & 5 \\ -2 & 2 & -6 & 0 & 0 \end{bmatrix}.$$

Ciclul, pe $\mathbf{X}^{(2)}$ (fig. 6.4, *c*) se construieşte pentru căsuţa (*3*, *3*), rezultînd $\mu = min\,\{4, 7, 2\} = 2$ şi o nouă soluţie $\mathbf{X}^{(3)}$ (v. fig. 6.4, *d*), care nu este cea optimală (**DIF** conţine valori negative).

$\boxed{k = 4}$

$$\begin{array}{l} u_1 + v_1 = 3 \\ u_1 + v_2 = 5 \\ u_1 + v_5 = 2 \\ u_2 + v_2 = 3 \\ u_2 + v_3 = 4 \\ u_3 + v_3 = 1 \\ u_3 + v_4 = 2 \end{array} \quad \rightarrow \quad \begin{array}{l} u_1 = 0 \\ u_2 = -2 \\ u_3 = -5 \end{array} \qquad \begin{array}{l} v_1 = 3 \\ v_2 = 5 \\ v_3 = 6 \\ v_4 = 7 \\ v_5 = 2 \end{array}$$

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} 3 & 5 & 6 & 7 & 2 \\ 1 & 3 & 4 & 5 & 0 \\ -2 & 0 & 1 & 2 & -3 \end{bmatrix}, \quad \mathbf{DIF} = \begin{bmatrix} 0 & 0 & -4 & -4 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 4 & 0 & 0 & 0 & 6 \end{bmatrix}.$$

Ciclul, pe $\mathbf{X}^{(3)}$ (fig. 6.4, *d*) se construieşte pentru căsuţa (*1*, *4*), rezultînd $\mu = min\,\{2, 5, 4\} = 2$ şi o nouă soluţie $\mathbf{X}^{(4)}$ (v. fig. 6.4, *e*), care se va îmbunătăţi printr-o nouă iteraţie:

$\boxed{k = 5}$

$$\begin{array}{l} u_1 + v_1 = 5 \\ u_1 + v_4 = 3 \\ u_1 + v_5 = 2 \\ u_2 + v_2 = 3 \\ u_2 + v_3 = 4 \\ u_3 + v_3 = 1 \\ u_3 + v_4 = 2 \end{array} \quad \rightarrow \quad \begin{array}{l} u_1 = 0 \\ u_2 = 2 \\ u_3 = -1 \end{array} \qquad \begin{array}{l} v_1 = 3 \\ v_2 = 1 \\ v_3 = 6 \\ v_4 = 3 \\ v_5 = 2 \end{array}$$

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} 3 & 1 & 2 & 3 & 2 \\ 5 & 3 & 4 & 5 & 4 \\ 2 & 0 & 1 & 2 & 1 \end{bmatrix}, \quad \mathbf{DIF} = \begin{bmatrix} 0 & 4 & 0 & 0 & 0 \\ -4 & 0 & 0 & 3 & 1 \\ 0 & 8 & 0 & 0 & 2 \end{bmatrix}.$$

Scanned with OKEN Scanner

Ciclul, pe $\mathbf{X}^{(4)}$ (fig. 6.4, *e*) se construieşte pentru căsuţa (*2*, *1*), rezultînd $\mu = min\{8, 3, 2\} = 2$ şi o nouă soluţie $\mathbf{X}^{(5)}$ (v. fig. 6.4, *f*) care se ameliorează printr-o nouă iteraţie.

$\boxed{k = 6}$

$$\begin{array}{l} u_1 + v_1 = 3 \\ u_1 + v_4 = 3 \\ u_1 + v_5 = 2 \\ u_2 + v_1 = 1 \\ u_2 + v_2 = 3 \\ u_2 + v_3 = 4 \\ u_3 + v_3 = 1 \end{array} \quad \rightarrow \quad \begin{array}{l} u_1 = 0 \\ u_2 = -2 \\ u_3 = -5 \end{array} \qquad \begin{array}{l} v_1 = 3 \\ v_2 = 5 \\ v_3 = 6 \\ v_4 = 3 \\ v_5 = 2 \end{array}$$

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} 3 & 5 & 6 & 3 & 2 \\ 1 & 3 & 4 & 1 & 0 \\ -2 & 0 & 1 & -2 & -3 \end{bmatrix}, \quad \mathbf{DIF} = \begin{bmatrix} 0 & 0 & -4 & 0 & 0 \\ 0 & 0 & 0 & 7 & 5 \\ 4 & 8 & 0 & 4 & 6 \end{bmatrix}.$$

Deoarece matricea **DIF** are un element negativ, din poziţia (*1*, *3*), $\mathbf{X}^{(5)}$ nu este soluţia optimală. Ea poate fi deci îmbunătăţită prin ciclul construit pentru căsuţa (*1*, *3*) din tabloul $\mathbf{X}^{(5)}$ (fig. 6.4, *f*), rezultînd $\mu = min\{6, 1\} = 1$ şi o nouă soluţie:

$$\mathbf{X}^{(6)} = \begin{bmatrix} 5 & 0 & 1 & 4 & 6 \\ 3 & 5 & 0 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 \end{bmatrix}.$$

pentru care:

$$\begin{array}{l} u_1 + v_1 = 3 \\ u_1 + v_3 = 2 \\ u_1 + v_4 = 3 \\ u_1 + v_5 = 2 \\ u_2 + v_1 = 1 \\ u_2 + v_2 = 3 \\ u_3 + v_3 = 1 \end{array} \quad \text{cu soluţiile} \quad \begin{array}{l} u_1 = 0 \\ u_2 = -2 \\ u_3 = -1 \end{array} \qquad \begin{array}{l} v_1 = 3 \\ v_2 = 1 \\ v_3 = 2 \\ v_4 = 3 \\ v_5 = 2 \end{array}$$

$$\mathbf{U} + \mathbf{V} = \begin{bmatrix} 3 & 1 & 2 & 3 & 2 \\ 1 & -1 & 0 & 1 & 0 \\ 2 & 0 & 1 & 2 & 1 \end{bmatrix} \text{ şi } \mathbf{DIF} = \begin{bmatrix} 0 & 4 & 0 & 0 & 0 \\ 0 & 4 & 4 & 7 & 5 \\ 0 & 8 & 0 & 0 & 2 \end{bmatrix}.$$

Deoarece **DIF** nu mai conţine nici o valoare negativă, rezultă că $\mathbf{X}^{(6)}$ este chiar soluţia optimală pentru care funcţia obiectiv ia valoarea minimă posibilă $F = 65$. Soluţia s-a obţinut, deci, după 6 iteraţii. Să observăm că ea diferă de soluţia obţinută pentru aceeaşi problemă la *Ex. 6—12* (matricea $\mathbf{X}^{(1)}$), însă rezultatul — în ceea ce priveşte valoarea minimă a funcţiei obiectiv şi restricţiile — este acelaşi.

*
* *

Scanned with OKEN Scanner

O altă situație specială are loc atunci cînd disponibilitățile (Σa) depășesc nevoile consumatorilor (Σb). În acest caz, se introduce un consumator fictiv (al $n+1$-lea) care solicită o cantitate de produse $b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$. Produsele care urmează, în urma rezolvării problemei, a fi livrate acestui consumator fictiv vor fi păstrate, ca rezerve, la furnizori.

Dacă disponibilitățile sînt mai mici decît nevoile consumatorilor, se introduce un expeditor fictiv, care să dispună — pentru livrare — de cantitatea de produse $a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i$.

În sfîrșit, dacă se impune condiția de a nu utiliza anumite rute, se atribuie acestor rute — în matricea **COST** — un cost de transport foarte mare, astfel încît pentru realizarea optimizării funcției obiectiv planul de transport să impună a nu fi folosite aceste rute prohibite sau interzise.

6.3. PACHETE DE PROGRAME SPECIALIZATE PENTRU REZOLVAREA PROBLEMELOR DE PROGRAMARE LINIARĂ

Programarea liniară a devenit o metodă consacrată de optimizare, frecvent utilizată în practică, în numeroase domenii ale economiei și tehnicii. Interesul față de această metodă este determinat astăzi exclusiv de eficiența utilizării programării liniare în activitatea curentă.

Deoarece în marea majoritate a aplicațiilor rezultă modele de mari dimensiuni, rezolvarea problemelor practice de programare liniară nu se poate realiza decît numai prin utilizarea tehnicilor moderne de calcul automat, care (așa cum am mai spus-o) au contribuit substanțial la proliferarea programării liniare în activitatea concretă.

Ca urmare, numeroasele probleme de programare liniară, ce sînt frecvent solicitate centrelor de calcul spre rezolvare, au determinat institutele de informatică, firmele constructoare și centrele de calcul să pună la punct și să dea în exploatarea curentă pachete de programe specializate pentru astfel de probleme, catalogate în sistemele de exploatare ale centrelor de calcul, care pot fi folosite cu ușurință de orice utilizator.

Din numeroasele pachete de programe ce pot fi utilizate la noi, am ales, pentru a le prezenta aici, pe două din cele mai reprezentative care acoperă un vast domeniu de aplicații și care pot fi întîlnite în toate centrele de calcul dotate cu sistemele de fabricație românească FELIX C 256, atît de răspîndite în țara noastră. Este vorba de programele de firmă TRANSPOR și OPALINE, care au fost realizate de fosta Compagnie Internationale pour l'Informatique (CII — „cedeuxi").

Scanned with OKEN Scanner

Program pentru problemele de transport — TRANSPOR

Modelul. Programul de firmă, din biblioteca de gestiune ştiinţifică CII, cu numele TRANSPOR este destinat rezolvării problemelor de transport care au modelele de forma:

$$\sum_{i=1}^{m} \cdot \sum_{j=1}^{n} c_{ij}x_{ij} = min \tag{6—51}$$

în condiţiile:

$$\sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1, 2, \ldots, m \tag{6—52}$$

$$\sum_{i=1}^{m} x_{ij} = b_j \qquad j = 1, 2, \ldots, n \tag{6—53}$$

$$x_{ij} \geqslant 0 \tag{6—54}$$

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j \tag{6—55}$$

$$a_i,\ b_j,\ c_{ij} \geqslant 0, \tag{6—56}$$

în care:

m reprezintă numărul punctelor-origine (sau puncte-sursă sau depozite de livrare sau, încă, liniile matricei cu costuri c_{ij});

n reprezintă numărul punctelor-destinaţie (sau staţii de consum sau, încă, coloanele matricei cu costuri c_{ij});

$a_1, a_2, \ldots, a_m$ — cantităţile de produse pe care le pot livra depozitele $1, 2, \ldots, m$;

$b_1, b_2, \ldots, b_m$ — cantităţile de produse pe care le solicită staţiile de consum $1, 2, \ldots, n$;

c_{ij} $(i = 1, 2, \ldots, m,\ j = 1, 2, \ldots, n)$ — costul transportului unei unităţi de produs de la depozitul i pînă la staţia de destinaţie j.

În acest fel, modelul de mai sus este un program liniar cu $m \times n$ variabile şi $m + n + 1$ restricţii. De fapt, observăm că ecuaţiile (6—52), (6—53) şi (6—55) nu sînt independente, ecuaţia (6—55) putînd fi dedusă din sistemul (6—52) şi (6—53); deci, numărul restricţiilor distincte este $m + n$. Faţă de un program liniar general, programul de transport prezintă particularitatea că datorită faptului că sistemul de restricţii (6—52), (6—53) şi (6—55) are matricea coeficienţilor numai cu valori 0 şi 1, determinantul matricelor de bază este egal fie cu $+1$, fie cu -1. Datorită acestui fapt, algoritmul simplex se reduce la un program foarte simplu şi foarte rapid, ceea ce permite rezolvarea fără dificultate a problemelor cu mari dimensiuni.

Din acest punct de vedere, limitările programului TRANSPOR sînt date de următoarele inegalităţi:

— pentru varianta care utilizează segmentul ALGO1 (ce reprezintă matricea costurilor prin valori întregi cu lungimea de un cuvînt $= 32$ bits):

$$4m + 4n + m \cdot n \leqslant T;$$

Scanned with OKEN Scanner

— pentru varianta care utilizează segmentul ALGO2 (ce reprezintă costurile c_{ij} în locații cu lungimea de o jumătate de cuvînt = 16 bits):

$$3m + 3n + \frac{1}{2} m \cdot n \leqslant T,$$

unde T reprezintă volumul din memoria internă a calculatorului (exprimat în octeți), disponibil pentru rezolvarea problemei. T se poate determina cu relația:

$$T = VMI - TP - TM,$$

unde VMI este volumul memoriei interne a calculatorului utilizat, TP — volumul ocupat în memoria internă de programul TRANSPOR (cel mult 2 500 cuvinte, adică $4 \times 2\,500 = 10\,000$ octeți) și TM este volumul ocupat în memoria internă de monitor.

Ex. 6—14. Să se determine limitele maxime (prin valorile m și n) ale unui program de transport în cazul unui sistem FELIX C 256 cu $VMI = 128$ kocteți și $TM = 28$ kocteți.

Rezultă:

$T = VMI - TP - TM = 128 \times 1\,024 - 10\,000 - 28 \times 1\,024 = 92\,400$ octeți;

— cu ALGO1

$4m + 4n + m \cdot n = 92\,400$ și dacă $m = 300$ rezultă $n = (92\,400 - 4m)/(4 + m) = (92\,400 - 1\,200)/304 = 300$;

— cu ALGO2

$3m + 3n + 0{,}5mn = 92\,400$ și dacă $m = 400$ rezultă $n = (92\,400 - 3m)/(3 + 0{,}5m) = (92\,400 - 1\,200)/203 = 400$. Deci probleme foarte mari (cu $m + n = 600$ la 800 restricții!).

Să notăm că ALGO2 este utilizat automat dacă ALGO1 provoacă o depășire a disponibilităților de memorie. Condițiile minimale la un sistem FELIX C 256 constau în: disponibilitățile de memorie să fie de minimum 32 767 octeți, iar coeficienții c_{ij} (costurile) să fie în număr de cel mult 5 000 (deci $m \cdot n < 5\,000$; de exemplu, la $m = 50$, $n \leqslant 100$). Precizăm că în aceste condiții minimale, la orice sistem FELIX C 256 nu este posibilă o depășire a volumului disponibil de memorie.

Utilizatorul mai trebuie să știe faptul că matricea costurilor c_{ij} nu trebuie să fie completă, adică nu este necesar ca toate punctele-origine să fie legate cu toate punctele de destinație. O pereche de puncte origine-destinație, între care nu sînt posibile transferuri de produse, va fi caracterizată de un cost „infinit".

Principiul algoritmului utilizat de TRANSPOR. Soluția de bază inițială este determinată prin metoda „drumului nord-vest, sud-est" (pe care am prezentat-o în subcapitolul precedent).

Îmbunătățirea unei soluții de bază și determinarea soluției optimale se face printr-un proces iterativ, bazat pe metoda clasică simplex, prin care sînt introduși și extrași din matricea de bază diverși vectori-coloană, în modul următor:

Scanned with OKEN Scanner

— calcularea așa-numitelor costuri reduse (pentru mulțimea variabilelor din afara bazei). Costurile reduse sînt valorile $c_{ij} - (u_i + v_j)$, adică elementele matricei pe care noi am notat-o cu **DIF** (v. *Ex. 6—12* și *Ex. 6—13*). Aceste costuri reduse se determină ca soluții ale unui sistem de ecuații liniare de forma $u_i + v_j = c_{ij}$ $i \in \{1, 2, \ldots, m\}, j \in \{1, 2, \ldots, \ldots, n\}$, care se rezolvă extrem de simplu, din aproape în aproape (v. *Ex. 6—12*, și *Ex. 6—13*);

— selectarea variabilei din afara bazei avînd costul redus „cel mai negativ" ca variabilă care va intra în noua bază. Dacă nu există astfel de variabile, care să poată intra în noua bază, deci dacă nu există costuri reduse negative, soluția cu care s-a început iterația este soluția optimală și calculul încetează;

— căutarea variabilei care va ieși din bază;

— schimbarea bazei și începerea unei noi iterații.

Organizarea generală a programului. Programul TRANSPOR este format dintr-un modul principal, denumit RACINE, care apelează succesiv diferitele segmente de care are nevoie și anume segmentele cu numele:

LECDO, ALGO1, ALGO2 și IMPRES.

Modulul principal RACINE, a cărui organigramă generală este reprezentată în figura 6.5, are în structura sa și următoarele subprograme: ENTETE, IMPITER, SPRINT și INIT.

Funcțiunile îndeplinite de aceste segmente și subprograme sînt, pe scurt, următoarele:

Modulul principal RACINE este programul director, care apelează în memorie (în mod „overlay") diferitele segmente ale programului (fig. 6.5). După faza de inițializare a variabilelor se introduce segmentul LECDO pentru citirea datelor. Apoi, în funcție de opțiunea utilizatorului, se trece la unul din segmentele ALGO1 sau ALGO2, care realizează algoritmul de prelucrare a datelor. Acest modul se termină prin apelarea segmentului IMPRES (fig. 6.5), pentru imprimarea rezultatelor finale. Modulul RACINE are și patru subprograme: ENTETE (care realizează imprimarea, la începutul paginii, a valorii inițiale a funcției obiectiv și — pe linia următoare — a capului relativ la iterații), IMPITER (care imprimă iterația curentă, adică: numărul iterației, numele liniei care intră în noua bază, valoarea sa la intrare, numele liniei care iese din bază și valoarea funcției obiectiv), SPRINT (înregistrează, plecînd dintr-o zonă memorie, un anumit număr de octeți editați conform unui format dat, însă în forma standard a imprimantei) și INIT (care permite ca, după introducerea programului în memorie, să se cunoască adresa primei locații libere și mărimea memoriei rămasă disponibilă).

Segmentul LECDO este utilizat pentru citirea mulțimii de date și pentru pregătirea tablourilor necesare calculului, conform algoritmului metodei. Este prevăzut, de la început, să se facă calculele cu ALGO1, adică cu locații de 4 octeți (ceea ce se realizează dîndu-se variabilei care face controlul selectării unuia din segmentele ALGO1 sau ALGO2 valoarea IALGO2 = 0); dar dacă volumul memoriei este insuficient, se trece la locații de 2 octeți (adică o jumătate de cuvînt), punîndu-se IALGO2 = 1,

Scanned with OKEN Scanner

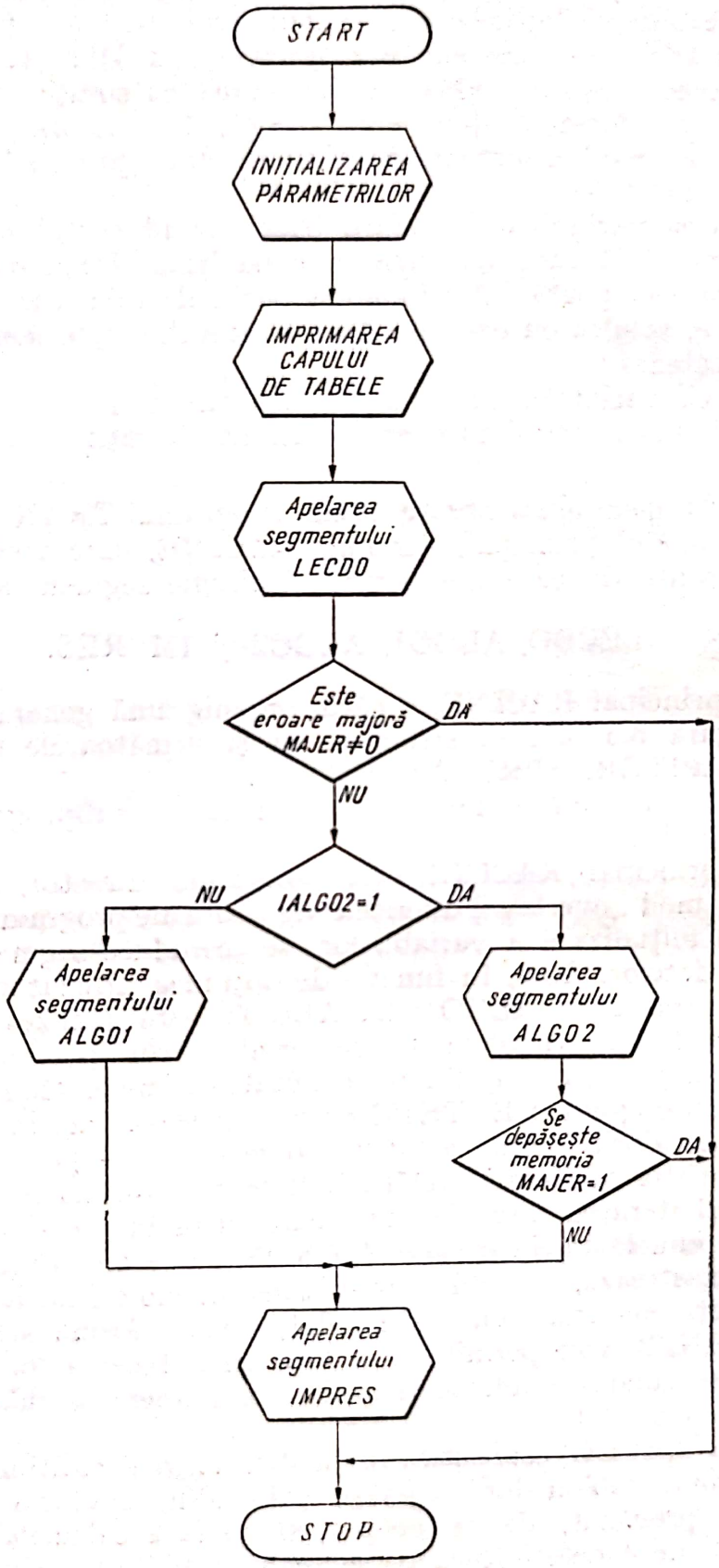


Fig. 6.5

Scanned with OKEN Scanner

ceea ce determină şi lucrul cu ALGO2 (v. fig. 6.5). Acest segment are 11 subprograme prin care se realizează manipularea dinamică a tablourilor de variabile.

Segmentul ALGO1 realizează prelucrarea datelor şi rezolvarea problemei cu metoda primal, matricea costurilor de transport fiind reprezentată prin valori întregi cu lungimea de un cuvînt (4 octeţi).

Segmentul ALGO2 este identic cu ALGO1, cu diferenţa (esenţială !) că matricea costurilor are elementele reprezentate prin 2 octeţi, toate calculele efectuîndu-se, deci, cu date de o jumătate de cuvînt.

Segmentul IMPRES realizează imprimarea, după găsirea soluţiei optimale, a variabilelor de bază cu valorile lor optimale, a valorii minime a funcţiei obiectiv şi a ultimei matrice cu costurile reduse (v. lista 6.5).

Prezentarea datelor problemei. Toate datele problemei sînt introduse în sistemul de calcul prin cartele perforate. Toate valorile numerice prelucrate de programele din TRANSPOR sînt de tip întreg. Pe cartele, datele trebuie aliniate la dreapta în zonele ce le sînt rezervate (v. fig.6.6).

Datele se prezintă grupat pe „capitole", fiecare capitol fiind definit printr-o cartelă „început-capitol", urmată de un anumit număr de cartele pe care sînt perforate datele corespunzătoare capitolului. Cartelele început-capitol cuprind următoarele nume: NAME, ORIGIN, DESTINATION, MATRIX şi ENDATA. Prezenţa tuturor acestor cartele este obligatorie, iar capitolele trebuie să fie aranjate strict în ordinea de mai sus (v. fig.6. 6).

Cartela NAME este prima din setul de cartele cu date şi are structura: cuvîntul NAME perforat pe coloanele 1 la 4, cuvîntul IMPR pe coloanele 9 la 12 (opţional, necesar a fi perforat numai dacă dorim imprimarea elementelor aferente iteraţiei în curs de execuţie) şi titlul programului, aşa cum este „compus" el de către utilizator, ce se perforează pe coloanele 13 la 80 (acest titlu va fi imprimat în capul paginii cu rezultatele problemei, aşa cum se arată, ca exemplu, în lista 6.5 de la *Ex. 6—15*). Zona coloanelor 13 la 80 de pe prima cartelă NAME poate fi lăsată „blanc" dacă nu vrem să se facă imprimarea vreunui titlu.

Capitolul ORIGIN cuprinde numele punctelor origine şi disponibilităţile de livrare (deci liniile tabloului de enunţare a problemei şi valorile a_i, $i = 1, 2, \ldots, m$). El începe cu o cartelă „început-capitol" care conţine numai numele ORIGIN perforat pe coloanele 1 la 6 şi continuă cu m cartele (cîte una pentru fiecare furnizor/linie), o astfel de cartelă conţinînd numele alfanumeric al punctului de origine perforat pe coloanele 5 la 12 şi valoarea disponibilităţii din acest punct origine, perforată pe coloanele 23 la 29, sub forma unui număr întreg aliniat la dreapta, pe coloana 29 (v. fig. 6.6).

Capitolul DESTINATION cuprinde numele punctelor destinaţie şi necesarul solicitat în aceste puncte (deci coloanele tabloului de enunţare a problemei şi valorile b_j, $j = 1, 2, \ldots, n$). Capitolul începe cu o cartelă „început-capitol" pe care este perforat numai cuvîntul DESTINATION pe coloanele 1 la 11 şi continuă cu n cartele (cîte una pentru fiecare consumator/coloană), o astfel de cartelă conţinînd numele alfanumeric al punctului de destinaţie perforat pe coloanele 5 la 12 şi valoarea necesarului din acest punct destinaţie, perforat pe coloanele 23 la 29, sub forma unui număr întreg aliniat la dreapta, pe coloana 29 (v. fig. 6.6).

Scanned with OKEN Scanner

Capitolul MATRIX introduce în calculator matricea costurilor unitare de transport c_{ij} ($i \in \{1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, n\}$) între punctele origine i şi punctele destinaţie j. Coeficienţii pot fi prezentaţi în orice ordine, aşa cum doreşte utilizatorul. Nu este necesar ca să dăm valori c_{ij} pentru toate perechile origine-destinaţie i—j; astfel elementele pentru care nu se introduc valori sînt considerate ca infinite (ceea ce înseamnă că se presupune că între punctele corespunzătoare costului infinit, transferul nu este posibil). Fiecare origine trebuie, însă, să fie „legată" printr-un cost finit de una sau mai multe destinaţii. Setul de cartele al acestui capitol este format din:

— o cartelă „început-capitol" care conţine numai numele MATRIX perforat pe coloanele 1 la 6;

— una sau mai multe cartele, şi anume maximum $m \cdot n$ pînă la $m \times (\text{întreg } \frac{n}{4} + 1)$ cartele (dacă n nu se divide cu 4) sau $m \cdot n/4$ (dacă n se împarte exact la 4), care conţin valorile c_{ij} şi poziţia lor i, j în matricea cost. Aceste cartele au structura următoare: pe coloanele 5 la 12 numele unui punct-origine (linie i), pe coloanele 15 la 22 numele unui punct-destinaţie (coloană j), pe coloanele 23 la 29 (aliniat pe 29) valoarea corespunzătoare liniei i şi coloanei j precedente a costului c_{ij}, pe coloanele 32 la 39 eventual numele altui punct-destinaţie (altă coloană j'), pe coloanele 40 la 46 (aliniat pe 46) valoarea costului $c_{ij'}$, pe coloanele 49 la 56 eventual numele altui punct-destinaţie (altă coloană j''), pe coloanele 57 la 63 (aliniat pe 63) valoarea costului $c_{ij''}$, pe coloanele 66 la 73 eventual numele altui punct-destinaţie (altă coloană j''') şi pe coloanele 74 la 80 (aliniat pe 80) valoarea costului $c_{ij'''}$ (v. fig. 6.6). Deci cartelele cu date ale capitoluului denumit MATRIX pot conţine 1 pînă la 4 coeficienţi c_{ij}, indicii i şi j fiind daţi prin numele punctelor origine (i) şi destinaţie (j).

Cartela ENDATA este, obligatoriu, cartela cu care se termină setul cartelelor cu date. Ea conţine numai numele ENDATA perforat pe coloanele 1 la 6 şi prezenţa ei determină începerea calculului de rezolvare a problemei.

Indicarea erorilor detectate. Pentru fiecare eroare detectată, programul imprimă conţinutul cartelei care a provocat detectarea erorii şi mesajul erorii. Astfel, erorile pot fi de trei feluri: erori majore care provoacă oprirea imediată a programului; erori majore care provoacă întreruperea programului numai după terminarea citirii datelor şi erori minore (care nu provoacă întreruperea programului).

Dacă nu sînt detectate erori, atunci se imprimă mesajul PAS D ERREUR MAJEUR iar dacă sînt, se indică unul din următoarele mesaje:

— în cazul erorilor majore care produc întreruperea imediată a programului;

DEBORDEMENT MEMOIRE (în cazul cînd memoria disponibilă a calculatorului nu este suficientă pentru rezolvarea problemei puse),

SOMME DEMANDES DIFFERENTE DE SOMME DISPONIBILITES (cînd sistemul de ecuaţii al restricţiilor este astfel formulat încît soluţia nu este realizabilă).

Scanned with OKEN Scanner

PAS DE CART NAME
PAS DE CARTE ORIGIN
PAS DE CARTE DESTINATION
PAS DE CARTE MATRIX
PAS DE CARTE ENDATA

toate acestea în cazul în care cartelele „început-capitol" au fost omise sau eronat perforate pe coloanele 1 la 4);

— în cazul erorilor majore care produc întreruperea după citirea datelor:

DISPONIBILITE OU DEMANDE NEGATIVE (atunci cînd un a_i sau/şi b_j a fost perforat cu semn minus pe cartelele din capitolele ORIGIN sau/şi DESTINATION),

COEFFICIENT DE COUT NEGATIF (cînd o valoare din cartelele capitolului MATRIX are semnul minus),

ERREUR DECODAGE VALEUR ENTIERE (cînd o zonă pentru date numerice din cartele conţine un caracter altul decît o cifră zecimală, blanc sau semn + sau —),

PAS DE COEF DANS MATRICE SUR LIGNE ⟨numele liniei⟩,

PAS DE COEF DANS MATRICE SUR COLONNE ⟨numele coloanei⟩ (cînd pe o linie sau/şi coloană nu e nici cel puţin un cost c_{ij} şi restricţia $\sum_i a_i = \sum_j b_j$ nu este satisfăcută);

— erori minore:

ORIGINE DUPLIQUEE (a doua cartelă origine este ignorată),

DESTINATION DUPLIQUEE (a doua cartelă destinaţie este ignorată),

ORIGINE INCONNUE SUR CARTE MATRICE (cartela matrice corespunzătoare este ignorată),

DESTINATION INCONNUE SUR CARTE MATRICE (coeficientul corespunzător unei destinaţii/coloană necunoscute pe cartelă este ignorat) şi

COEFFICIENT DEJA INITIALISE (se ignoră o nouă valoare a coeficientului).

Rezultatele prelucrării. Rezultatele programului TRANSPOR sînt extrase din sistem prin intermediul imprimantei rapide. Rezultatele obţinute cu acest program, de pe „listing", sînt următoarele:

— în cursul citirii datelor: erorile detectate;

— după terminarea citirii datelor: segmentul de program ales pentru efectuarea calculelor de găsire a soluţiei optimale, adică ALGO1 sau ALGO2, prin tipărirea inducţiei DEBUT OPTIMISATION (ALGO1) sau DEBUT OPTIMISATION (ALGO2) (v. lista 6.5). Atragem atenţia cititorului că în cazul utilizării segmentului ALGO2, dacă există trasee $i—j$ interzise (pentru care nu s-a indicat costul c_{ij} şi astfel acesta a fost considerat infinit), s-ar putea ca valorile considerate de ALGO2 ca fiind „infinite" (şi anume 32 767) să nu fie suficient de mari în raport cu celelalte date ale problemei şi, ca urmare, să se introducă în bază şi cuplul $i—j$ nedorit. De aceea utilizatorul va trebui să verifice cu atenţie soluţiile obţinute în acest caz;

— în cursul efectuării iteraţiilor de găsire a soluţiei optimale (şi numai dacă pe coloanele 9 la 12 ale cartelei NAME utilizatorul a perforat

Scanned with OKEN Scanner

cuvîntul IMPR): numărul iteraţiei, perechea numelor origine-destinaţie care indică variabila ce intră în bază, valoarea acestei variabile, perechea de nume origine-destinaţie ce definesc variabila care iese din bază şi valoarea funcţiei obiectiv după executarea iteraţiei curente (v. lista 6.5);

— după găsirea soluţiei optimale (deci după terminarea calculelor): lista variabilelor de bază (fiecare variabilă este definită prin perechea de nume origine-destinaţie), numărul total de iteraţii, valoarea (minimă posibilă) a funcţiei obiectiv şi matricea costurilor reduse imprimată linie cu linie (v. lista 6.5). Deoarece soluţia optimală a fost găsită (altminteri nu s-ar fi făcut această imprimare!), elementele din matricea costurilor reduse nu pot fi decît pozitive sau nule (pentru variabilele din soluţia de bază mai mari decît zero).

Utilizarea practică. Programul TRANSPOR este stocat pe bandă magnetică sub forma imaginii disc a unei biblioteci IMT. Deci, pentru a putea fi utilizat, trebuie duplicată această bandă pe disc cu ajutorul programului BIBLIOTHECAIRE.

La marea majoritate a centrelor de calcul dotate cu sisteme FELIX C 256, programul TRANSPOR este deja transferat pe un disc magnetic (pe care, de obicei, sînt înregistrate pachetele de programe: MATHLIB, GESTOCK şi PROGLIN, acesta din urmă conţinînd şi programul TRANSPOR). În acest caz, se cere operatorului montarea fizică a pachetului de discuri ce conţine biblioteca de programe liniare PROGLIN (de exemplu, discul cu numărul 33) pe o anumită unitate periferică de discuri şi, printr-o cartelă de comandă de tip FETCH, se aduce în memoria internă programul TRANSPOR din biblioteca PROGLIN (v. lista 6.5).

Pachetul de cartele al utilizatorului va avea, în acest caz, următoarea structură:

```
coloana 1      coloana 11
   ↓              ↓
   .           JOB xxxxxxxx,AN:xxxx,PN:xxxxxxxx
   .           ** MONTAŢI DISCUL xx PE RDx
   .           STANDBY
   .           FETCH DV:RDx,LN:PROGLIN,FN:TRANSPOR, GN:1
               VN:1, UN:1
   .           RUN
datele problemei
   .           EOJ
```

Ex. 6—15. Pentru a ilustra modul cum se utilizează programul de bibliotecă TRANSPOR, vom considera o problemă de transport (şi anume problema din *Ex. 6—4*) pe care o vom rezolva cu ajutorul TRANSPOR-ului.

Notînd punctele origine (liniile) cu DEPOZIT1, DEPOZIT2 şi DEPOZIT3 (în loc de *D1*, *D2* şi *D3* cum sînt indicate în tabela 6—4 din *Ex. 6—4* şi aceasta numai pentru a exemplifica — credem noi mai elocvent — posibilităţile de identificare a liniilor şi coloanelor în cadrul programului TRANSPOR) şi punctele destinaţie (coloanele) cu STATIA1, STATIA2,

..., STATIA5 (în loc de *S1*, *S2*, ..., *S5* cum sînt numite în tabela 6—4 din *Ex. 6—4*), datele problemei, așa cum au fost formulate în exemplul 6—4, sînt următoarele:

Tabela costurilor unitare de transport c_{ij}

punctele destinație (coloane) / punctele origine (linii)	STAȚIA 1	STAȚIA 2	STAȚIA 3	STAȚIA 4	STAȚIA 5
DEPOZIT1	3	5	2	3	2
DEPOZIT2	1	3	4	8	5
DEPOZIT3	2	8	1	2	3

Disponibilitățile în punctele origine (a_i)

DEPOZIT1	16
DEPOZIT2	8
DEPOZIT3	6

Necesarul la destinații (b_j)

STATIA1	8
STATIA2	5
STATIA3	7
STATIA4	4
STATIA5	6

Aceste date se perforează pe cartele așa cum este indicat în formularul din figura 6.6, unde se poate vedea și conținutul celorlalte cartele cu directive de comandă.

Rezultatele, așa cum au fost tipărite de imprimanta rapidă, sînt redate în lista 6.5. După cum se vede, s-a obținut aceeași soluție ca la *Ex. 6—13*

Programul OPALINE

OPALINE (OPtimisation Accélérée de programmes LINEaires) este un program de firmă (CII) destinat rezolvării problemelor de optimizare reprezentate prin modele liniare conținînd pînă la 1 500 restricții, în funcție de configurația sistemului de calcul utilizat.

Modelul matematic al problemelor pe care le poate rezolva programul OPALINE este de forma:
să se găsească vectorul coloană **X**, cu dimensiunea n', astfel încît:

$$\mathbf{CX} = minim/maxim \qquad (6—57)$$

$$\mathbf{A}_1\mathbf{X} = \mathbf{B}_1 \qquad (6—58)$$

$$\mathbf{A}_2\mathbf{X} \geqslant \mathbf{B}_2 \qquad (6—59)$$

$$\mathbf{A}_3\mathbf{X} \leqslant \mathbf{B}_3 \qquad (6—60)$$

$$\mathbf{B}_4 \leqslant \mathbf{A}_4\mathbf{X} \leqslant \mathbf{B}'_4 \qquad (6—61)$$

$$x_i \text{ limitat } i \in \{1, 2, \ldots, n'\} \qquad (6—62)$$

unde:
C este un vector linie cu n' elemente (coeficienții funcției obiectiv);

Scanned with OKEN Scanner

IPG PLOIEȘTI

Formular de programare

PROGRAM OPTITRAN		INSTRUCȚIUNI DE PERFORARE	Simbolul	ϕ							Pag 1 Din
PROGRAMATOR : ION DUMITRESCU	Data 16.VI.1975		Se perforează	ϕ							

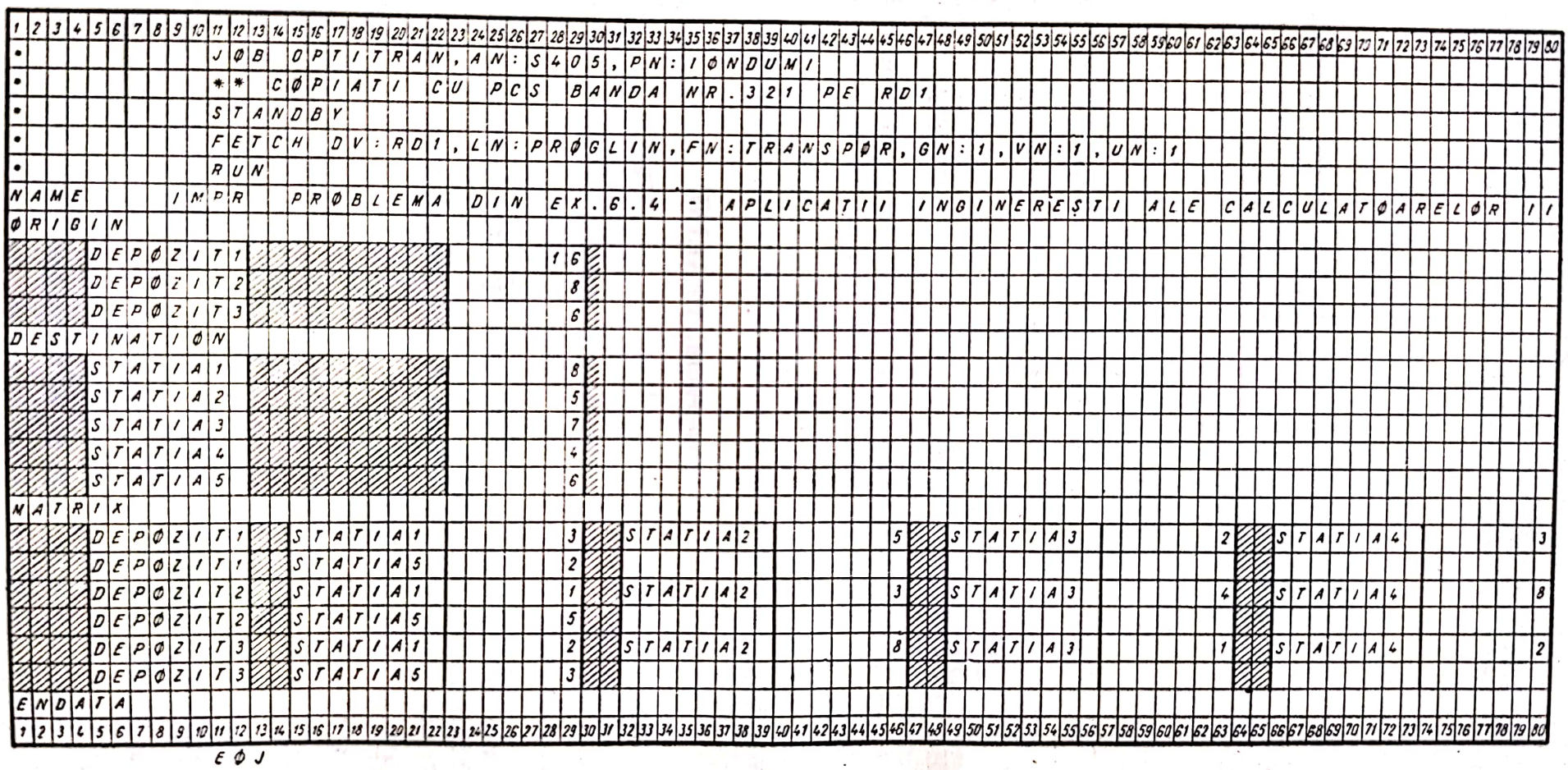


Fig. 6.6

```
.          JOB OPTITRAN,AC:S405,PN:UNIBUC1
.          ** MONTATI BANDA PILOT LA INRAMANTA
.          ** COPIATI CU PCS BANDA NR.321 PE RD1
.          STANDBY
.          FETCH DV:RD1,LN:PROGLIN,FN:TRANSPOR,GN:1,VN:1,UN:4
.          RUN
TRANSPOR STARTED
```

** PROGRAMME DE TRANSPORT C.I.I. **

DEBUT LECTURE DES DONNEES

PAS D ERREUR MAJEURE

DEBUT OPTIMISATION (ALGO1)

VALEUR INITIALE DE LA FONCTION : 121

ITERATION	ENTRANT DANS LA BASE	VALEUR A L ENTREE	SORTANT DE LA BASE	VALEUR DE LA FONCTION
1	DEPOZIT1 - STATIA5	3	DEPOZIT1 - STATIA3	106
2	DEPOZIT2 - STATIA2	1	DEPOZIT2 - STATIA4	97
3	DEPOZIT3 - STATIA5	2	DEPOZIT3 - STATIA5	85
4	DEPOZIT1 - STATIA4	2	DEPOZIT1 - STATIA2	77
5	DEPOZIT2 - STATIA1	2	DEPOZIT3 - STATIA4	69
6	DEPOZIT1 - STATIA3	1	DEPOZIT2 - STATIA3	65

Scanned with OKEN Scanner

** PROBLEMA DIN EX.6.4 - APLICATII INGINERESTI ALE CALCULATOATELOR II **

SOLUTION OPTIMALE

LISTE DES VARIABLES DE BASE

ORIGINE	DESTINATION	VALEUR
DEPOZIT1	STATIA1	5
DEPOZIT1	STATIA3	1
DEPOZIT1	STATIA4	4
DEPOZIT1	STATIA5	6
DEPOZIT2	STATIA1	3
DEPOZIT2	STATIA2	5
DEPOZIT3	STATIA3	6

NOMBRE TOTAL D ITERATIONS : 6

VALEUR DE LA FONCTION 65

MATRICE DES COUTS REDUITS (IMPRIMEE PAR LIGNES)

DEPOZIT1	0	0	0	0	0
DEPOZIT2	0	0	4	7	5
DEPOZIT3	0	4	0	0	2

FIN DE PROGRAMME

Lista 6.5

Scanned with OKEN Scanner

$\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$ și $\mathbf{A}_4$ sînt matrice cu n' coloane și m_1, m_2, m_3 — respectiv — m_4 linii ale căror elemente sînt coeficienții necunoscutelor x din restricțiile modelului;

$\mathbf{B}_1$, $\mathbf{B}_2$, $\mathbf{B}_3$ și $\mathbf{B}_4$ sînt vectori coloană cu m_1, m_2, m_3 și —respectiv— m_4 elemente reprezentate de termenii liberi ai restricțiilor.

Condițiile de tip (6—62), de limitare a variabilelor, pot lua una din următoarele forme:

$x_i \geqslant 0$ — condiția de nenegativitate;
$x_i \geqslant \alpha_i$ — variabile limitate inferior,
$x_i \leqslant \beta_i$ — variabile limitate superior,
$\alpha_i \leqslant x_i \leqslant \beta_i$ — variabile dublu limitate,
x_i oarecare — variabile libere,

unde $i \in \{1, 2, \ldots, n'\}$.

Modelul de programare liniară prezentat sub forma (6—57), (6—58), ..., (6—62) reprezintă „formularea externă" a problemei (externă din punctul de vedere al calculatorului); această formulare corespunde întocmai modului cum sînt acceptate datele unei probleme de programare liniară de către OPALINE. De aceea, utilizatorul trebuie să rețină că orice problemă de programare liniară pe care vrea să o rezolve cu ajutorul OPALINE-ului trebuie să fie formulată în forma externă (6—57), ..., (6—62).

Pentru rezolvarea problemelor de forma (6—57), ... (6—62) prin algoritmul simplex, OPALINE-ul folosește o „reprezentare internă" sub forma canonică:

$$\begin{aligned} &\mathbf{CX} = minim/maxim \\ &\mathbf{AX} = \mathbf{B} \\ &x_i \geqslant 0 \\ &0 \leqslant x_i \leqslant \beta_i \quad i \in \{1, 2, \ldots, n\} \end{aligned} \tag{6—63}$$

unde: $\mathbf{C}$ este un vector linie cu n elemente, $\mathbf{A}$ este o matrice cu m linii și n coloane, $\mathbf{B}$ este un vector coloană cu m termeni, iar $\mathbf{X}$ este vectorul coloană al soluției (cu n componente).

Repetăm faptul că utilizatorul nu este direct interesat de formularea internă (6—63) deoarece citirea tuturor datelor, precum și imprimarea rezultatelor finale, se face exclusiv în formularea externă*.

Caracteristicile generale ale programului OPALINE. Acest program este realizat în întregime în limbajul de asamblare specific sistemului IRIS (Limbajul ASSIRIS) și reprezintă, în fapt, un cod pentru rezolvarea problemelor de programare liniară pe baza metodei primale revizuite a algoritmului simplex, cu utilizarea formei de produs a matricelor cu inversare implicită a bazei [34].

În principiu, numărul maxim (teoretic) al restricțiilor ce pot fi „tratate" este de 4 095, iar numărul variabilelor este limitat numai de faptul

* Cunoașterea formulării interne poate prezenta interes numai pentru a înțelege anumite rezultate intermediare, din cursul procesului de calcul (de optimizare a soluției din iterația precedentă celei în curs).

Scanned with OKEN Scanner

că numărul total al termenilor nenuli ai matricei **A** din (6—63) trebuie să fie mai mic decît 32 768.

OPALINE-ul are o structură modulară, fiind constituit dintr-un modul principal şi din 8 segmente care asigură următoarele funcţii:

— citirea datelor problemei şi creerea de fişiere standard interne (segmentele denumite prin LECDO1, LECDO2 şi LECDO3);

— calculul iterativ de îmbunătăţire a soluţiei (optimizarea problemei) prin segmentele cu numele: INVI şi ALGO1 sau ALGO2;

— transpunerea rezultatelor în reprezentarea externă şi imprimarea lor (prin segmentele denumite IMP1 şi IMP2).

OPALINE-ul creează două fişiere: unul cu acces direct pe disc DIAM şi altul fie cu acces direct pe disc DIAM, fie nedefinit cu organizare standard pe bandă magnetică (la alegerea utilizatorului).

Utilizarea OPALINE-ului necesită un sistem de calcul FELIX (IRIS) cu următoarea configuraţie minimală:

— unitate centrală FELIX C 256,

— o unitate disc DIAM,

— eventual un derulator de bandă magnetică (dacă utilizatorul cere creerea celui de al doilea fişier pe bandă magnetică);

— un cititor de cartele perforate,

— o imprimantă rapidă.

Performanţele OPALINE-ului depind direct de disponibilităţile de memorie internă, din cauza numeroaselor masive (tablouri) cu care lucrează. Disponibilităţile de memorie internă, în măsura în care conduc la frecvente „consultări" ale fişierelor din memoria externă, determină în bună parte şi timpul de calcul.

În ceea ce priveşte fişierele create de OPALINE, fişierul pe disc DIAM este un fişier de lucru, temporar, care conţine matricea de optimizare, coeficienţii funcţiei obiectiv şi limitele variabilelor (necesită 14 octeţi/variabilă pentru datele iniţiale şi 6 octeţi/coeficient nenul al matricei în curs de optimizare). Fişierul pe disc sau bandă (la alegere, printr-un parametru indicat de utilizator în prima cartelă cu date) este creat de cele trei segmente de citire a datelor (LECDO1, LECDO2 şi LECDO3) şi conţine toate datele iniţiale sub o formă intermediară, precum şi baza curentă din timpul calculului (acest al doilea fişier este un fişier de „păstrare" şi de „reluare" pentru o nouă rezolvare a problemei).

Dimensiunile problemei ce poate fi rezolvată cu OPALINE-ul depind de configuraţia sistemului de calcul (în special prin dimensiunile memoriei interne şi, eventual, ale discului DIAM).

Limitările referitoare la memorie trebuie verificate în prealabil de către utilizator, pentru a vedea dacă problema sa (pe care o are de soluţionat) poate fi rezolvată cu calculatorul la care apelează. Verificările de „dimensiuni" se fac aşa cum se arată în continuare.

1° În faza de citire:

1°1. Limitări ale memoriei interne (principale). Ele sînt de două feluri; una depinde de structura programului şi este independentă de spaţiul disponibil în memorie, iar cealaltă — din contră — depinde de disponibilităţile de memorie. Depăşirea memoriei se poate produce în diferite momente ale fazei de citire a datelor. Trebuie verificate următoarele condiţii:

Scanned with OKEN Scanner

— numărul total al termenilor nenuli (fie în matricea inițială, fie în cea finală) să nu depășească 32 767;

— să fie îndeplinită următoarea condiție referitoare la disponibilul de memorie:

$$4{,}5NVE + 3{,}5NR + 2{,}25NLC + 1{,}5NLIM < D - BUF,$$

în care:

$D = DM - DS - DP$ (DM — dimensiunile memoriei interne în cuvinte de cîte 4 octeți, DS — dimensiunile memoriei ocupată de monitor și DP — dimensiunile memoriei ocupată de programul OPALINE, care este de 7 140 cuvinte = 28 560 octeți), $BUF = 768$ cuvinte = 3072 octeți ce reprezintă dimensiunile tamponului de intrare/ieșire ale programului OPALINE, NVE — numărul de variabile efective ale problemei inițiale (în reprezentarea externă, adică n' din relațiile 6—57, ..., 6—62), NR — numărul de restricții al problemei, NLC — numărul de linii citite (adică numărul de restricții + numărul funcțiilor libere) și $NLIM$ — numărul de variabile structurale limitate superior.

1°2. Limitări din punctul de vedere al discului DIAM. Pentru a putea stoca fișierul temporar, volumul necesar pe discul DIAM (în octeți) trebuie să satisfacă relația:

$$6NTT + 14N < (NSDIAM - 4) \times VS,$$

în care: VS reprezintă volumul sectorului de disc ($VS = 1\,088$ octeți), $NSDIAM$ — numărul sectoarelor disponibile pe DIAM, N — numărul de variabile în matricea finală a modelului de programare liniară; NTT — numărul total de termeni nenuli din matricea finală. Aceste limitări, cu privire la dimensiunile memoriei pe discul DIAM, nu sînt în general critice.

1°3. Dacă utilizatorul alege, ca suport al fișierului de păstrare, banda magnetică, atunci singura condiție impusă pentru aceasta este ca ea să fie o bandă monovolum (fișierul se înregistrează pe o singură bandă fizică, de pe o singură rolă).

2° În faza de calcul (de optimizare a soluției):

Această fază nu pune nici o problemă în ceea ce privește discul DIAM (deoarece acest disc nu este utilizat decît la citirea matricei finale). Deci, dacă dimensiunile discului DIAM au corespuns în faza de citire, ele vor corespunde și în faza de calcul a soluției optime.

În această fază este utilizată memoria internă și dimensiunile acestei memorii trebuie să fie suficiente pentru a putea conține în întregime matricea inversă a modelului, ceea ce implică satisfacerea următoarei condiții:

$$6{,}5M + N + 2{,}5NTT + 1{,}5NLIM < D - BUF,$$

unde notațiile au semnificațiile arătate mai sus (la care adăugăm că M reprezintă numărul total de linii în matricea finală). Deoarece în formula de mai sus, pentru o problemă dată, toți termenii sînt cunoscuți în afară de NTT, înseamnă că ea ne permite să determinăm o valoare limită superioară a lui NTT (numărul total al termenilor pe care îi poate avea, în timpul calculului, matricea inversă).

Scanned with OKEN Scanner

Dacă se utilizează în OPALINE segmentul ALGO1 (în locul lui ALGO2), cînd matricea inițială se introduce tot în memoria centrală, trebuie respectată condiția:

$$6{,}5\ M + N + 2{,}5NTT + 1{,}5NTNI + 1{,}5NLIM < D,$$

unde $NTNI$ reprezintă numărul de termeni nenuli din matricea inițială ce trebuie optimizată (inclusiv variabilele de ecart).

În cazul depășirii memoriei, este inutil să se mai reia executarea programului, deoarece memoria nu are dimensiunile necesare problemei de rezolvat.

3° în faza de editare:

În această fază trebuie ca, din punctul de vedere al memoriei interne (centrale), să fie satisfăcută condiția:

$$3{,}5NR + 3NVE < D - BUF,$$

iar din punctul de vedere al discului DIAM, condiția:

$$6NTNI + 22\ N + 8M < (\text{NSDIA}M - 8) \times VS(\text{în octeți}).$$

Prezentarea datelor. Datele unei probleme de programare liniară, rezolvată cu ajutorul OPALINE-ului, se introduc în sistemul de calcul prin cartele perforate. Datele sînt organizate sub forma unor „capitole", fiecare capitol fiind format dintr-un grup de cartele perforate (prima cartelă a unui grup este cartela care definește capitolul și poartă numele de cartelă-comandă, iar restul de cartele, ce urmează după prima, sînt cartelele cu datele efective ale problemei corespunzătoare capitolului).

În cazul în care programul OPALINE este utilizat în „mod creație", adică la rezolvarea „de la început" a unei probleme, cu toate datele sale,* *„capitolele" cu datele de intrare* sînt, în ordine, următoarele: NAME urmat de OPTIONS urmat obligatoriu de ROWS urmat obligatoriu de COLUMNS urmat de RHS urmat de RANGES urmat de BOUNDS urmat obligatoriu de ENDATA urmat obligatoriu de MINI sau MAXI urmat de BASIS și încheiat obligatoriu de START. Absența, din cartelele cu date, a unui capitol indicat mai sus ca necesitînd o prezență obligatorie, determină oprirea programului.

1° Capitolul NAME este format dintr-o singură cartelă care are următoarea structură:

— pe coloanele 1 la 4 se perforează cuvîntul NAME

— pe coloana 8 se perforează litera B dacă utilizatorul dorește ca fișierul de păstrare să fie pe bandă sau „blanc" (nici o perforație) dacă utilizatorul folosește discul DIAM pentru fișierul de păstrare

— pe coloanele 9 la 12 se perforează un număr cuprins între — 999 și 9999 prin care utilizatorul poate să-și identifice problema (în „modul creație" de utilizare a OPALINE-ului acest număr nu este necesar)

* OPALINE-ul mai poate fi utilizat și în „modul reluare", prin care se continuă rezolvarea aceleiași probleme sau a unei probleme similare dedusă din cea inițială (cu anumite modificări relative la funcția obiectiv sau la limitele variabilelor). În această prezentare a OPALINE-ului nu am avut în vedere decît utilizarea în „mod creație".

Scanned with OKEN Scanner

— pe coloanele 13 la 80 se perforează titlul programului (prin cel mult 68 de caractere alfanumerice), care va fi reprodus în capul fiecărei pagini a „listing"-ului cu rezultate.

În modul de utilizare „creație", cartela NAME nu este obligatorie; în lipsa ei (sau dacă pe coloanele 13 la 80 ale cartelei NAME nu se indică nici o denumire), titlul imprimat pe „listing" va fi SIMPLEXE CII OPALINE

2° Capitolul OPTIONS, care este facultativ, se utilizează numai pentru a modifica anumiți parametri ai rulării prin OPALINE care, altminteri, au anumite valori standard. Acest capitol poate fi format din cartela-comandă, care are structura:

— pe coloanele 1 la 7 se perforează cuvîntul OPTIONS

și din 1 la 12 cartele cu valorile impuse parametrilor de rulare, care au structura:

— pe coloanele 1 la 4 se perforează numele parametrului

— pe coloanele 17 la 20 se perforează valoarea impusă de utilizator parametrului (în afară de cartela LIST pe care, pe coloanele 17 la 20, se lasă blanc).

Parametrii de rulare și valorile lor standard (considerate din oficiu dacă nu se introduc cartele de OPTIONS) sînt următorii:

ITER (numărul maxim de iterații) 1 000;

REIN (frecvența maximă a reinversărilor) 50;

FREQ (frecvența imprimării iterațiilor în curs de execuție) 0 (nici una). Dacă utilizatorul dorește să știe cum evoluează problema de la o iterație la alta, poate solicita (printr-o cartelă FREQ) 1 pînă la 10 000 de imprimări intermediare;

SOUS (numărul de coloane multiple alese ca vectori de intrare în bază) 1;

CYCL (numărul maxim de iterații înaintea unei ciclări);

PIVM (valoarea limită sub care în cursul optimizării un pivot, luat în valoare absolută, este considerat ca fiind nul) — 6 (ceea ce reprezintă valoarea 10^{-6});

TZER (valoarea limită sub care depășirea limitei unei variabile este considerată nulă) — 7 adică 10^{-7};

TCOS (toleranța pentru care un termen $c_j - F_j$/v. **Metoda simplex.** p. 33/ este considerat negativ) — 5 adică 10^{-5};

EPSM (toleranța în calculul vîrfurilor soluției; dacă $\mathbf{A}+\mathbf{B} < \text{EPSM} \times |\mathbf{A}|$ vîrful $\mathbf{A}+\mathbf{B}$ este anulat) — 10 adică 10^{-10};

TECO (valoarea sub care orice termen calculat este considerat nul) — 12 adică 10^{-12};

TPIV (valoarea limită sub care în cursul inversării cu pivot, luat în valoare absolută, este considerat nul) — 6 adică 10^{-6};

LIST (prezența acestei cartele, într-un capitol OPTIONS, face ca în faza de citire a datelor să se imprime lista variabilelor și a restricțiilor).

3° Capitolul ROWS, care este obligatoriu, definește liniile modelului problemei (funcția obiectiv și restricțiile). El este format din cartela — comandă (prima), care are structura:

Scanned with OKEN Scanner

— pe coloanele 1 la 4 perforat cuvîntul ROWS

şi din cîte o cartelă pentru fiecare linie a problemei, care are structura:

— pe coloanele 2 şi 3 simbolul naturii liniei

— pe coloanele 5 la 12 numele (alfanumeric) al liniei aşa cum îl alege utilizatorul.

Simbolul care indică natura liniei modelului este:

N pentru funcţia liberă (aşa cum este orice funcţie obiectiv),
G pentru restricţiile de tipul $\geqslant$ (mai mare sau egal),
L pentru restricţiile de tipul $\leqslant$ (mai mic sau egal),
E pentru restricţiile de tipul $=$ (egal).

4° Capitolul COLUMNS, care este obligatoriu, defineşte variabilele naturale (din modelul în formă externă) şi conţine valorile coeficienţilor matricei modelului (numai valorile nenule) în ordinea „coloană după coloană". El este format din cartela-comandă (prima), care are structura:

— pe coloanele 1 la 7 perforat cuvîntul COLUMNS

şi cartele cu valorile coeficienţilor (una sau două valori pe o cartelă), care au structura:

— pe coloanele 5 la 12 perforat numele (alfanumeric) al coloanei

— pe coloanele 15 la 12 perforat numele unei linii (aşa cum a fost definit în capitolul ROWS)

— pe coloanele 25 la 36 perforată valoarea (în format F) a coeficientului care se găseşte pe coloana şi linia definite anterior, pe coloanele 5—12 (coloana) şi 15—22 (linia)

— facultativ, pe coloanele 40 la 47 perforat numele altei linii

— facultativ, pe coloanele 50 la 61 perforată valoarea (în format F) a coeficientului aflat pe coloana definită pe cîmpul 5—12 şi linia din cîmpul 40—47.

Se introduc atîtea cartele (după cartela-comandă COLUMNS) cîte sînt necesare ca să prezentăm toţi coeficienţii nenuli ai modelului. Denumirea coloanelor (prin 1 la 8 caractere alfanumerice) se face de către utilizator.

5° Capitolul RHS, care este facultativ, se utilizează pentru prezentarea valorilor termenilor din membrul drept al restricţiilor (elementele matricei **B**). Sînt perforaţi pe cartelele RHS numai termenii nenuli din membrul drept. Prima cartelă a acestui capitol este cartela-comandă cu structura:

— pe coloanele 1 la 3 perforat cuvîntul RHS

care este urmată de atîtea cartele cîte sînt necesare să se indice toţi coeficienţii nenuli din membrul drept al restricţiilor, perforaţi cîte unul (sau doi) pe cartele cu structura:

— pe coloanele 5 la 12 perforat numele alfanumeric dat membrului din dreapta al restricţiilor

— pe coloanele 15 la 22 perforat numele unei linii (aşa cum a fost definit în capitolul ROWS)

— pe coloanele 25 la 36 perforată valoarea coeficientului din membrul drept aflat pe linia indicată în cîmpul 15—22

— facultativ, pe coloanele 40 la 47 perforat numele altei linii

Scanned with OKEN Scanner

— facultativ, pe coloanele 50 la 61 perforată valoarea coeficientului din membrul drept aflat pe linia definită în cîmpul 40—47.

Dacă capitolul RHS lipseşte din cartelele cu date, se consideră din oficiu că membrul din dreapta al restricţiilor este format numai din termeni nuli.

6° Capitolul RANGES, care este facultativ, defineşte (dacă este prezent) restricţiile cu două limite simultane (inferioară şi superioară). Prima cartelă a acestui capitol este cartela-comandă care are structura:

— pe coloanele 1 la 6 perforat cuvîntul RANGES

după care urmează cartelele cu valorile de ecart (una sau, facultativ, două valori pe o cartelă) de forma:

— pe coloanele 5 la 12 perforat numele grupului RANGES (acest nume este, însă, ignorat de program)

— pe coloanele 15 la 22 numele liniei restricţie cu două limite (aşa cum este definit în capitolul ROWS)

— pe coloanele 25 la 36 perforată valoarea ecartului (în format *F*)

— facultativ, pe coloanele 40 la 47 perforat numele altei linii restricţie cu două limite

— facultativ, pe coloanele 50 la 61 perforată valoarea ecartului corespunzător liniei indicată în cîmpul 40—47.

7° Capitolul BOUNDS, care este facultativ, defineşte (dacă este prezent) limitele unor variabile ale modelului.

Să reţinem că variabilele sînt presupuse, implicit, pozitive sau nule (deci cu limita „mai mare sau egal cu zero"); de aceea, numai pentru variabilele care au alte limite (decît cea considerată implicit) acestea trebuie definite printr-un grup de limite în capitolul BOUNDS.

Structura capitolului este următoarea: o cartelă-comandă avînd perforat pe coloanele 1 la 6 cuvîntul BOUNDS urmată de una sau mai multe cartele cu valorile limitelor, fiecare avînd:

— pe coloanele 2 şi 3 perforat tipul limitei (adică unul din cuvintele: LO pentru $>$, UP pentru $<$, FX pentru variabilele fixe sau FR pentru variabilele libere)

— pe coloanele 5 la 12 perforat numele grupului de limite (un cuvînt cu 1 la 8 caractere alfanumerice)

— pe coloanele 15 la 22 perforat numele coloanei (aşa cum a fost definit în capitolul COLUMNS) corespunzător variabilei considerate

— pe coloanele 25 la 36 perforată valoarea limitei (în format F).

Scanned with OKEN Scanner

În cazul unei variabile libere (tipul FR), valoarea perforată pe coloanele 25 la 36 este ignorată, deoarece în acest caz ea nu are nici o semnificaţie.

8° Cartela ENDATA (a cărei prezenţă este obligatorie) indică sfîrşitul pachetului de cartele cu datele problemei. Ea conţine cuvîntul ENDATA perforat pe coloanele 1 la 6.

9° Cartela pentru indicarea modului de optimizare prin funcţia obiectiv (adică minimizare sau maximizare) este o cartelă obligatorie şi are următoarea structură:

— pe coloanele 1 la 4 perforat unul din cuvintele MINI sau MAXI

— pe coloanele 9 la 16 perforat numele funcţiei de optimizat (aşa cum a fost definit în capitolul ROWS)

— pe coloanele 21 la 28 perforat numele membrului din dreapta al restricţiilor (cel definit în capitolul RHS)

— pe coloanele 33 la 40 perforat numele grupului de limite (corespunzător numelui folosit în capitolul BOUNDS).

10° Capitolul BASIS, care este facultativ, permite să se definească coloanele ce vor fi considerate iniţial ca bază. El este format din cartela-comandă (avînd cuvîntul BASIS perforat pe coloanele 1 la 5) şi una sau mai multe cartele care conţin variabilele ce intră în baza iniţială (o variabilă, perforată pe coloanele 3 la 10, sau facultativ două pînă la opt variabile pe cîte o cartelă, a două variabilă ce intră iniţial în bază perforîndu-se pe coloanele 13 la 20, a treia variabilă — pe coloanele 23 la 30, ... şi a opta variabilă — pe coloanele 73 la 80).

Dacă capitolul BASIS lipseşte, atunci baza iniţială este aleasă automat de programul OPALINE (prin modulul său de „crashing").

11° Cartela START, a cărei prezenţă este obligatorie pentru a comanda începerea operaţiilor de optimizare a funcţiei obiectiv, conţine cuvîntul START perforat pe coloanele 1 la 5.

Modul de utilizare. La majoritatea centrelor de calcul care sînt dotate cu sisteme FELIX C 256, programul OPALINE este catalogat în biblioteca PROGLIN pe un disc magnetic; de aceea, pentru rezolvarea unei probleme de programare liniară, mai întîi trebuie apelat în memoria centrală programul OPALINE. Aceasta se face cu ajutorul unor directive (cartele) de control care preced pachetul de cartele cu date. Prin aceste directive se asignează şi cele două fişiere utilizate de program:

— fişierul de „păstrare" (pentru datele şi rezultatele finale) IDX K denumit OPAL, care poate avea ca suport discul magnetic sau banda magnetică (la alegerea utilizatorului, ceea ce înseamnă că există două variante de cartele de control);

— fişierul de lucru (pentru stocarea intermediară a rezultatelor, necesare calculului iterativ) IDX L cu numele TRAV situat (obligatoriu) pe disc magnetic.

Scanned with OKEN Scanner

Astfel, în cazul folosirii discului ca suport pentru OPAL (fișierul de „păstrare") cartelele utilizatorului au structura:

```
coloana 1    coloana 11
   ↓            ↓
   .            JOB OPALINE, AN :xxxx,PN :xxxxxxxx
   .            ** MONTAȚI DISCUL xxx PE RDm
   .            STANDBY
   .            FETCH DV :RDm,LN : PROGLIN,FN :OPALINE, GN :1,
   .            VN :1,UN :1
   .            INIT DV :RDn,VS :000001
   .            ALLOC DV : RDn,FN :'OPAL', SZ :ij,AM :ANY      } disc
   .            ASSIGN K,DV :RDn
   .            LABEL K,FN :'OPAL',AM :OFL
   .            INIT DV :RDp,VS :000001
   .            ALLOC DV :RDp,FN :'TRAV',SZ :ij,AM :ANY
   .            ASSIGN L, DV :RDp
   .            LABEL L,FN :'TRAV',AM :OFL
   .            RUN

   NAME      }
   .         }
   .         } datele problemei
   .         }
   START     }
   .            EOJ
```

În cazul folosirii benzii magnetice ca suport pentru OPAL, cartelele de control marcate mai sus cu } *disc* se înlocuiesc cu cartelele:

```
coloana1     coloana 11
   ↓            ↓
   .            INIT DV :MTn,VS :000001      }
   .            ASSIGN K,DV :MTn             } bandă
   .            LABEL K,FN :'OPAL',AM :OFL   }
```

În cele de mai sus, simbolurile *x* reprezintă caractere alfanumerice alese de utilizator, iar *m*, *n*, *p*, *i* și *j* reprezintă cifre zecimale (*m*, *n* și *p* numerotează unitatea periferică de memorie externă, iar *i* și *j* numărul de cilindri rezervați pentru fișier pe disc).

Modul de prezentare a rezultatelor, precum și unele din mesajele sistemului cu privire la lucrarea efectuată, pot fi deduse din lista 6.6 obținută pentru problema din *Ex. 6—16*, exemplu cu care încheiem acest capitol.

Ex. 6—16. Vom ilustra modul cum se utilizează practic programul OPALINE folosindu-l la rezolvarea problemei din *Ex. 6—5*.

Scanned with OKEN Scanner

IPG PLOIESTI

Formular de programare

PROGRAM ØPALINE		INSTRUCTIUNI DE PERFORARE	Simbolul	Ø						Pag 1 Din
PROGRAMATOR IØN DUMITRESCU IPG	Data 25 VII 1975		Se perforează	0						

```
•         JØB ØPALINE,AN·S405.PN·IØNDUMI
•         ** MØNTATI BANDA PILØT LA IMPRIMANTA
•         ** MØNTATI PE RD1 DISCUL 33 CØNTININD BIBLIØTECA PRØGLIN
•         STANDBY
•         FETCH DV RD1,LN·PRØGLIN,FN·ØPALINE.GN 1,VN 1,UN 1
•         INIT DV RD2,VS 000001
•         ALLØC DV·RD2.FN 'ØPAL'.SZ:12.AM ANY
•         ASSIGN K,DV·RD2
•         LABEL K.FN 'ØPAL'.AM ØFL
•         INIT DV RD3,VS 000001
•         ALLØC DV RD3,FN 'TRAV'.SZ·12,AM ANY
•         ASSIGN L.DV RD3
•         LABEL L.FN 'TRAV'.AM ØFL
•         RUN
NAME        REZØLVAREA PRØBLEMEI DIN EX.6-5 PRIN UTILIZAREA ØPALINE-ULUI
ØPTIØNS             1
FREQ
LIST
RØWS
 N  ØBIECTIV
 G  RESTR  1
 L  RESTR  2
 L  RESTR  3
 L  RESTR  4
```

IPC PLOIEȘTI

Formular de programare

PROGRAM : ØPALINE		INSTRUCȚIUNI DE PERFORARE	Simbolul	Ø							Pag. 2 Din
PROGRAMATOR : IØN DUMITRESCU - IPG	Data 25.VII.1975		Se perforează	0							

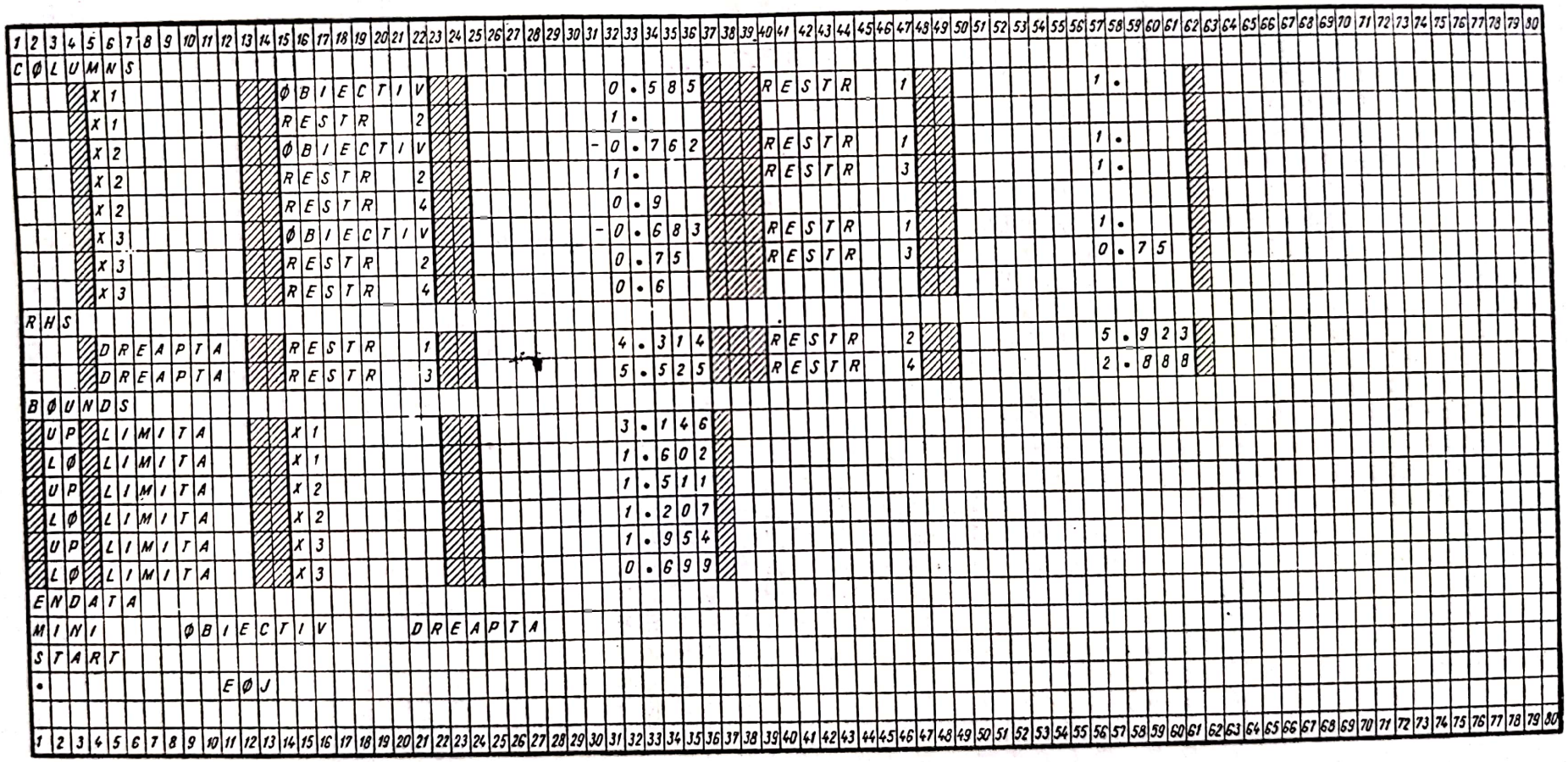


Fig. 6.7

Scanned with OKEN Scanner

Potrivit relaţiilor (6—33) şi (6—34), modelul pe baza căruia se va face optimizarea este următorul:

$$0{,}585.x_1 - 0{,}762.x_2 - 0{,}683.x_3 = min \quad (0)$$
$$1 .x_1 + 1 .x_2 + 1 .x_3 \geqslant 4{,}314 \quad (1)$$
$$1 .x_1 + 1 .x_2 + 0{,}75 .x_3 \leqslant 5{,}923 \quad (2)$$
$$1 .x_2 + 0{,}75 .x_3 \leqslant 5{,}525 \quad (3)$$
$$0{,}9 .x_2 + 0{,}6 .x_3 \leqslant 2{,}888 \quad (4)$$

la care se adaugă şi condiţiile:

$$x_1 \leqslant 3{,}146 \qquad x_2 \leqslant 1{,}511 \qquad x_3 \leqslant 1{,}954 \quad (5)$$
$$x_1 \geqslant 1{,}602 \qquad x_2 \geqslant 1{,}207 \qquad x_3 \geqslant 0{,}699$$

Deci, în forma sa externă, modelul are cinci linii: funcţia *obiectiv* (0), *restricţia* (1), *restricţia* (2), ..., *restricţia* (4) şi *limitele* (5), precum şi trei coloane (potrivit necunoscutelor x_1, x_2 şi x_3) şi coloana membrului din *dreapta*. Acest model conţine toate datele de intrare (datele problemei), care vor fi perforate pe cartele aşa cum rezultă din formularele prezentate în figura 6.7 (care reprezintă, în ordine, toate cartelele necesare pentru utilizarea OPALINE-ului la rezolvarea problemei de mai sus). Pentru desemnarea liniilor, coloanelor şi condiţiilor modelului (0), (1), ..., (5) am utilizat numele: OBIECTIV, RESTR 1, RESTR 2, ..., RESTR 4 (pentru linii, la capitolul ROWS), X1, X2 şi X3 (pentru coloane, la capitolul COLUMNS), DREAPTA (pentru termenul din dreapta al modelului, la capitolul RHS) şi LIMITA (pentru condiţiile (5), la capitolul BOUNDS). Aceste denumiri pot fi alese de către utilizator şi altfel (noi le-am dedus din denumirile generale ale modelului, cele scrise cursiv mai sus).

Rezultatele problemei, aşa cum au fost tipărite de imprimantă, sînt prezentate în lista 6.6. Soluţiile optimale (extrase cu numai 3 cifre la partea fracţionară) sînt:

$$x_1 = 1{,}602$$
$$x_2 = 1{,}511$$
$$x_3 = 1{,}954$$

pentru care minimul funcţiei obiectiv este $-1{,}549$.

Ţinînd seama de expresiile (0) de mai sus şi (6—32) şi (6—33) din *Ex. 6—5*, rezultă că mărimile care descriu procesul de strunjire a filetului la sapele de foraj cu trei role au următoarele valori optimale (în condiţiile problemei):

$$n = \text{alg } x_1 = 40 \text{ rot/min},$$

$$t = \frac{1}{100} \text{alg } x_2 = 0{,}324 \text{ mm},$$

$$s = \frac{1}{100} \text{alg } x_3 = 0{,}9 \text{ mm/rot},$$

pentru care cheltuielile au valoarea minimă, adică $c_t = \text{alg } c = \text{alg } (-1{,}549 - 0{,}098) = 0{,}444 < 0{,}50$ lei/sapă (deci, costul total al filetării este de 444 lei la o mie de sape).

Scanned with OKEN Scanner

```
.          JOB OPALINE,AN:S405,PN:IONDUMI
.          ** MONTATI BANDA PILOT LA IMPRIMANTA
.          ** MONTATI PE RD1 DISCUL 33 CONTININD BIBLIOTECA PROGLIN
.          STANDBY
.          FETCH DV:RD1,LN:PROGLIN,FN:OPALINE,GN:1,VN:1,UN:1
.          INIT DV:RD2,VS:000001
.          ALLOC DV:RD2,FN:'OPAL',SZ:12,AM:ANY
.          ASSIGN K,DV:RD2
.          LABEL K,FN:'OPAL',AM:OFL
.          INIT DV:RD3,VS:000001
.          ALLOC DV:RD3,FN:'TRAV',SZ:12,AM:ANY
.          ASSIGN L,DV:RD3
.          LABEL L,FN:'TRAV',AM:OFL
.          RUN
OPALINE  STARTED

                                REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

 DEBUT LECTURE DES DONNEES


                                REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI


                                        MATRICE INITIALE


                                NB TOTAL LIGNES                5
                                  NB FONCTIONS                 1
                                  NB TOTAL CONTRAINTES         4
                                     EGALITES                  0
                                     INEGALITES INF            3
                                     INEGALITES SUP            1


                                NB VARIABLES EFFECTIVES        3
                                REMPL MAT BRUTE (/)        66.67
```

Lista 6.6

REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

LIGNES

SYMBOLES	NUMEROS ASSOCIES	NATURE
OBJECTIV	1	N
RESTR 1	2	G
RESTR 2	3	L
RESTR 3	4	L
RESTR 4	5	L

REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

COLONNES

SYMBOLES	NUMEROS ASSOCIES	NB TERMES NON NULS
X1	1	3
X2	2	5
X3	3	5

REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

PAS D ERREUR MAJEURE

Scanned with OKEN Scanner

REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

DONNEES ET PARAMETRES RELATIFS A

FONCT ECON	OBJECTIV
SEC MEMBRE	DREAPTA
BORNES	LIMITA

NB TOTAL DE LIGNES	5	NB TOTAL DE COLONNES	
FONCTION ECONOMIQUE	1	VARIABLES EFFECTIVES	
CONTRAINTES EFFECTIVES	4	VAR EFF BORN SUP	
CONTRAINTES INEGALITE	4	VAR ECART (CONT INEGAL)	
CONTRAINTES EGALITE	0	VAR ECART (BORN SUP)	

MATRICE FINALE - NB TERMES NON NULS 17
- REMPLISSAGE (%) 48.57

SUPPORT FICHIER :	DISQUE
NB MAX ITERATIONS	1000
FREQ REINVERSION	50
FREQ IMPRESSION	1
NB ITER AV CYCLAGE	100
NB SOUS OPTIM	1
LIMITE-PIVOT	0.10E-05
LIMITE-VAR	0.10E-06
LIMITE-NT	0.10E-04
LIMITE-SOMME	0.10E-09
LIMITE-VAL DP	0.10E-11
LIMITE-PIV REL	0.10E-11

REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

DEBUT INVERSION NB TERMES= 0
FIN INVERSION NB TERMES= 0 F OBJ= -0.4599874D 00

	ITER	X ARTI	VEC ENT	VEC SOR	F OBJ
*	1	1	1	0	0.7152868D-01

FIN PHASE I

	ITER	DJ NEG	VEC ENT	VEC SOR	F OBJ
*	2	2	3	1	-0.1010479D 01
*	3	2	4	3	-0.1317146D 01
*	4	1	2	2	-0.1548794D 01

DEBUT INVERSION NB TERMES= 13
FIN INVERSION NB TERMES= 0 F OBJ= -0.1548795D 01

Scanned with OKEN Scanner

REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

RESULTATS RELATIFS A

FONCT ECON	OBJECTIV
SEC MEMBRE	DREAPTA
BORNES	LINITA

SOLUTION OPTIMALE

PREC EGAL	- MAX	0.510-06
	- SOM	0.110-05
PREC DJ B	- MAX	0.000 00
	- SOM	0.000 00
Nb D ITERATIONS		4
NB REINVERSIONS		2

Scanned with OKEN Scanner

REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

VARIABLES

NUMERO	SYMBOLE	ETAT	VALEUR	COEF-COUT	BORNE INF	BORNE SUP	COUT-REU
1	X1	NB	0.16019993D 01	0.58499998E 00	0.16019993E 01	0.31459995E 01	0.58499998E 00
2	X2	NB	0.15109997D 01	-0.76200002E 00	0.12069998E 01	0.15109997E 01	-0.76200002E 00
3	X3	NB	0.19539995D 01	-0.68300003E 00	0.69899994E 00	0.19539995E 01	-0.68300003E 00

REZOLVAREA PROBLEMEI DIN EX.6-5 PRIN UTILIZAREA OPALINE-ULUI

CONTRAINTES

NUMERO	SYMBOLE	ETAT	VALEUR	BORNE INF	BORNE SUP	VALEUR DUALE
1	RESTR 1	BS	0.50649985D 01	0.43139992E 01	AUCUNE	0.00000000E 00
2	RESTR 2	BS	0.45784986D 01	AUCUNE	0.59229994E 01	0.00000000E 00
3	RESTR 3	BS	0.29764993D 01	AUCUNE	0.55249996E 01	0.00000000E 00
4	RESTR 4	BS	0.25329993D 01	AUCUNE	0.28879995E 01	0.00000000E 00

FONCTION -0.15487940D 01

Scanned with OKEN Scanner

Capitolul 7

APLICAŢIILE PROGRAMĂRII NELINIARE

O problemă de programare neliniară, denumită *program neliniar*, este o problemă de optimizare în care intervin modele neliniare, adică reprezentări matematice unde cel puţin o funcţie nu este liniară.

Acest capitol a fost conceput în aşa fel încît să se arate cum se formulează un program neliniar aferent unor activităţi inginereşti practice şi cum poate fi el soluţionat, în mod efectiv, prin utilizarea tehnicilor de calcul automat de care dispunem în prezent.

7.1. PROGRAME NELINIARE

După cum se ştie, în orice activitate inginerească, o problemă de optimizare se formulează pe baza următoarelor elemente:

— un sistem ce trebuie condus;
— un obiectiv ce trebuie atins cu ajutorul sistemului;
— o mulţime de comenzi admise;
— o măsură a eficienţei de realizare a obiectivului urmărit.

Un proces chimic constituie un exemplu important de sistem ce trebuie condus. În reactorul chimic din figura 7.1, *a* se presupune că se desfăşoară

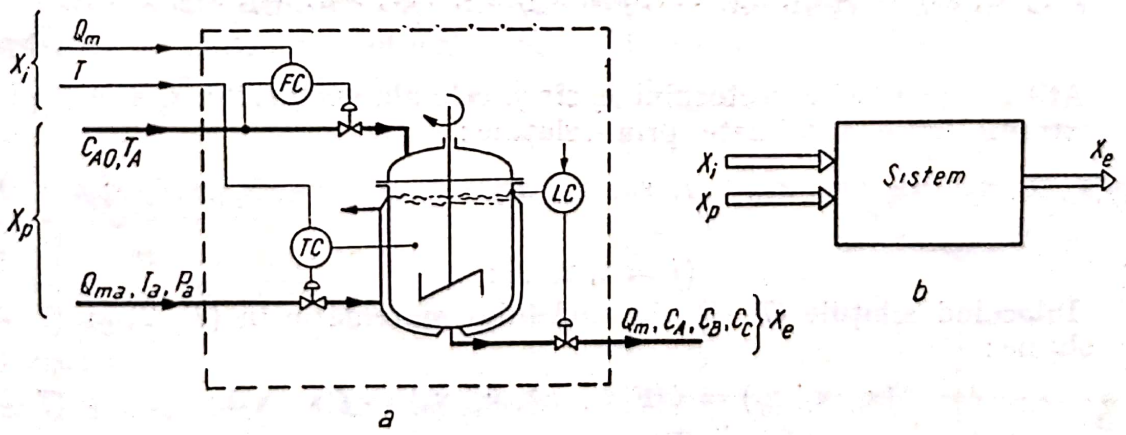


Fig .7.1

reacţia $A \rightarrow B \rightarrow C$. Produsul B care se obţine constituie obiectivul urmărit. Obţinerea acestui produs la nivelul calitativ şi cantitativ dorit se realizează prin modificarea „într-un mod corespunzător" a debitului de materie primă Q_m şi a temperaturii din reactor T, care în cazul de faţă constituie comenzile U. Debitul de materie primă şi temperatura nu pot lua orice valori, ele sînt supuse unor restricţii atît în ceea ce priveşte valorile minime cît şi cele maxime.

Obţinerea produsului B trebuie însoţită de anumite condiţii de eficienţă tehnică şi economică. Un exemplu în acest sens poate să îl reprezinte preţul de cost care trebuie să fie minim, sau cantitatea din produsul principal B, care trebuie să fie maximă.

Să presupunem că s-au determinat valorile debitului de materie primă şi temperaturii pentru care produsul principal B se obţine la condiţiile de calitate impuse şi cu un preţ de cost minim. Aceste valori au rezultat pentru anumite valori ale perturbaţiilor: compoziţia şi temperatura materiei prime, debitul, temperatura şi presiunea agentului termic etc. Atunci cînd valorile acestor perturbaţii se modifică, rezultă o nouă stare a sistemului care, bineînţeles, nu va mai fi optimă. În condiţiile altor valori ale perturbaţiilor trebuie determinate noi valori pentru comenzi, astfel încît preţul de cost să fie din nou minim.

Punerea şi rezolvarea problemei în acest mod are semnificaţia unui optim în regim static; avem de-a face cu o *problemă de optimizare statică.*

Sistemele fizice pot fi reprezentate formal ca în figura 7.1, *b*, unde $\mathbf{x}_i$ reprezintă vectorul comandă, m-dimensional, $\mathbf{x}_p$ — vectorul perturbaţie, p-dimensional şi $\mathbf{x}_e$ — vectorul ieşire, n-dimensional. Modelul matematic în regim static, pentru un asemenea sistem, se exprimă cu ajutorul relaţiei vectoriale:

$$\mathbf{x}_e = \mathbf{F}_i(\mathbf{x}_i, \mathbf{x}_p),$$

sau relaţiilor scalare:

$$x_{ek} = F_k(x_{i1}, x_{i2}, \ldots, x_{im}, x_{p1}, x_{p2}, \ldots, x_{pp}), \qquad (7-1)$$
$$(k = 1, 2, \ldots, n)$$

iar măsura de eficienţă a conducerii procesului prin funcţia:

$$\psi(\mathbf{x}_e, \mathbf{x}_i, \mathbf{x}_p) = \psi(x_{e1}, x_{e2}, \ldots, x_{en}, x_{i1}, x_{i2}, \ldots, x_{im}, x_{p1}, x_{p2}, \ldots x_{pp}). \qquad (7-2)$$

Atît componentele vectorului $\mathbf{x}_e$ cît şi cele ale vectorului $\mathbf{x}_i$ sînt supuse la restricţii fizice exprimate prin relaţiile:

$$\varphi_j(\mathbf{x}_e, \mathbf{x}_i, \mathbf{x}_p) = \varphi_j(x_{e1}, x_{e2}, \ldots, x_{en}, x_{i1}, x_{i2}, \ldots, x_{im}, x_{p1}, x_{p2}, \ldots, x_{pp}) \leqslant 0. \qquad (7-3)$$
$$(j = 1, 2, \ldots, l)$$

Înlocuind relaţiile (7—1) ale modelului matematic în (7—2) şi (7—3) se obţine:

$$\psi(\mathbf{x}_e, \mathbf{x}_i, \mathbf{x}_p) = \psi(\mathbf{F}(\mathbf{x}_i, \mathbf{x}_p), \mathbf{x}_i, \mathbf{x}_p) = f(\mathbf{x}_i, \mathbf{x}_p); \qquad (7-4)$$

$$\varphi_j(\mathbf{x}_e, \mathbf{x}_i, \mathbf{x}_p) = \varphi_j(\mathbf{F}(\mathbf{x}_i, \mathbf{x}_p), \mathbf{x}_i, \mathbf{x}_p) = g_j(\mathbf{x}_i, \mathbf{x}_p) \leqslant 0. \qquad (7-5)$$
$$(j = 1, 2, \ldots, l)$$

Scanned with OKEN Scanner

Cu aceasta, problema de optimizare în regim static a sistemului din figura 7.1, *b* poate fi formulată astfel:

Să se determine minimul (maximul) funcției (7—4) în condițiile existenței restricțiilor (7—5).

Observăm că atît funcția de eficiență (7—4), cît și restricțiile (7—5) conțin ca variabile independente componentele vectorului $\mathbf{x}_i$ și vectorului perturbației $\mathbf{x}_p$. Diferența dintre componentele vectorului comandă $\mathbf{x}_i$ și componentele vectorului perturbație $\mathbf{x}_p$ constă în aceea că asupra primelor se poate acționa, modificîndu-le după necesitățile operatorului. Asupra componentelor vectorului $\mathbf{x}_p$ nu se poate interveni în mod deliberat; ele pot fi în cazul cel mai bun măsurate. Există desigur și componente ale vectorului $\mathbf{x}_p$ care nu pot fi măsurate, dar în cele ce urmează nu vom lua în discuție acest caz.

Rezolvarea problemei de optim [(7—4), (7—5)] se efectuează în felul următor:

1° pentru anumite valori ale componentelor vectorului $\mathbf{x}_p$, funcțiile f și g_j vor depinde numai de componentele vectorului $\mathbf{x}_i$:

$$min(max)\ f(\mathbf{x}_i) \qquad (7—6)$$

$$\mathbf{g}(\mathbf{x}_i) \leqslant 0; \qquad (7—7)$$

2° se rezolvă problema de optim (7—6), (7—7), obținîndu-se extremul funcției f în condițiile prezenței restricțiilor g_j;

3° atunci cînd componentele vectorului $\mathbf{x}_p$ iau alte valori, se repetă punctele 1° și 2°.

În figura 7.2 este ilustrat cazul unei probleme de minim la care vectorii $\mathbf{x}_i$ și $\mathbf{x}_p$ sînt unidimensionali, iar restricțiile $g_j(x_i)$ de forma $x_{im} \leqslant x_i \leqslant x_{iM}$. Pentru cazul $\mathbf{x}_p = \mathbf{x}_{po}$, minimul funcției de eficiență este f_0, deci coincide cu minimul problemei fără restricție, în timp ce pentru $\mathbf{x}_p = \mathbf{x}_{p1}$, minimul este f_R, diferit de minimul fără restricții f_L.

În cazul al doilea, soluția problemei este limitată de valoarea maximă admisă pentru comandă $\mathbf{x}_i = \mathbf{x}_{iM}$. Minimul f_R este un *optim cu restricții*, în timp ce minimul f_L este un *optim liber*.

În cele ce urmează se vor avea în vedere probleme de optimizare aduse la forma (7—6) și (7—7). Pentru simplificarea notațiilor, vectorul $\mathbf{x}_i$ va fi notat prin $\mathbf{x}$. Deoarece:

$$max\, f(\mathbf{x}) = -\, min\, [-f(\mathbf{x})], \qquad (7—8)$$

rezultă că oricare problemă de maximizare poate fi pusă sub forma unei probleme de minimizare.

De obicei variabilele $x_1, x_2, \ldots, x_m$ sînt supuse și condițiilor de pozitivitate:

$$x_1 \geqslant 0, \quad x_2 \geqslant 0, \ldots, x_m \geqslant 0, \qquad (7—9)$$

acestea rezultînd din sensul lor fizic în probleme concrete.

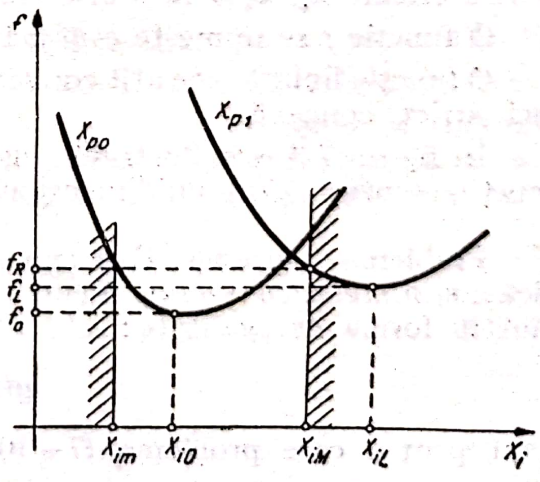


Fig. 7.2

Scanned with OKEN Scanner

Cu aceste observaţii, problema [(7—6, (7—7)] se scrie simbolic sub forma:

$$min\{f(\mathbf{x}) \mid \mathbf{g}(\mathbf{x}) \leqslant 0,\ \mathbf{x} \geqslant 0\}. \qquad (7-10)$$

Soluţia problemei (7—10) se va nota cu $\mathbf{x}$, iar valoarea minimă a funcţiei de eficienţă $f(\mathbf{x})$.

Deoarece restricţiile $\mathbf{g}(\mathbf{x}) \leqslant 0$ şi funcţia de eficienţă $f(\mathbf{x})$ sînt neliniare, problema (7—10) este denumită *problemă de programare neliniară* (sau, pe scurt, *program neliniar*). Pentru probleme de programare neliniară puse sub această formă foarte generală nu se cunosc, pînă în prezent, metode generale de rezolvare. Rezultate bune s-au obţinut pentru subclasa problemelor (7—10), la care funcţiile $g_j(\mathbf{x})$, $j = 1, 2, \ldots, l$ şi $f(\mathbf{x})$ sînt *funcţii convexe*.

Astfel de probleme sînt cunoscute sub numele de *programe convexe*.

Un caz aparte îl constituie subclasa problemelor (7—10), la care funcţiile $g_j(\mathbf{x})$ sînt *liniare*, iar funcţia de eficienţă $f(\mathbf{x})$ este *pătratică*. Asemenea probleme sînt numite *programe pătratice*. Desigur că programarea pătratică este un caz particular al programării convexe, constituind un domeniu în care s-au obţinut în ultimii ani rezultate fundamentale.

Deoarece conceptul de *convexitate* are o importanţă deosebită pentru problemele ce vor fi expuse în continuare, vom reaminti cititorului noţiunile de *mulţime convexă* şi *funcţie convexă*.

Definiţia 1. Fie punctele $\mathbf{x}_1, \mathbf{x}_2 \in R^n$. Mulţimea $[\mathbf{x}_1, \mathbf{x}_2]$ se numeşte segment dacă:

$$[\mathbf{x}_1, \mathbf{x}_2] = \{\mathbf{x} \in R^n | \mathbf{x} = \lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2,\ 0 \leqslant \lambda \leqslant 1\}. \qquad (7-11)$$

Mulţimea $\mathbf{X} \subset R^n$ se numeşte convexă dacă pentru oricare două puncte $\mathbf{x}_1, \mathbf{x}_2 \in R^n$ există $[\mathbf{x}_1, \mathbf{x}_2] \subset \mathbf{X}$.

Prin $[\mathbf{x}_1, \mathbf{x}_2]$ s-a notat segmentul care uneşte punctele $\mathbf{x}_1$ şi $\mathbf{x}_2$.

Definiţia 2. Fie mulţimea convexă $\mathbf{X} \subset R^n$. Funcţia $f: \mathbf{X} \to R$ se numeşte convexă dacă:

$$f(\lambda\mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2) \leqslant \lambda f(\mathbf{x}_1) + (1 - \lambda)f(\mathbf{x}_2), \qquad (7-12)$$

pentru oricare $\mathbf{x}_1, \mathbf{x}_2 \in \mathbf{X}$ şi orice λ, $0 \leqslant \lambda \leqslant 1$.

O funcţie f se numeşte *concavă* dacă $-f$ este o funcţie convexă.

O funcţie liniară este atît convexă cît şi concavă, însă nici strict convexă, nici strict concavă.

În figura 7.3 este ilustrată sugestiv proprietatea de convexitate a unei funcţii, pentru cazul unidimensional.

Problema Lagrange. Este cunoscută şi sub denumirea de problema clasică a *minimului cu legături*. Pentru acest gen de probleme, restricţiile (7—7) sînt în forma de egalitate:

$$\mathbf{g}(\mathbf{x}) = 0, \qquad (7-13)$$

fapt pentru care problema (7—10) devine:

$$min\{f(\mathbf{x}) \mid \mathbf{g}(\mathbf{x}) = 0\}. \qquad (7-14)$$

Scanned with OKEN Scanner

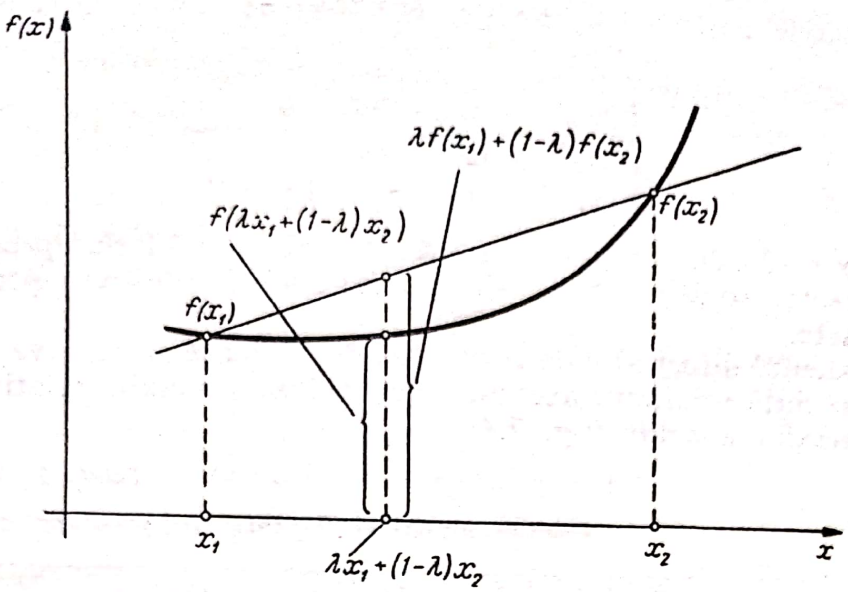


Fig. 7.3

Se presupune că funcţiile f şi g_j, $j = 1, 2, \ldots, l$ admit derivate de ordinul întîi în raport cu toate variabilele x_i, $i = 1, 2, \ldots, m$.

Pentru rezolvarea problemei de minim (7—14) se alcătuieşte funcţia $F : R^{l+m} \to R$ (denumită şi lagrangean):

$$F(\mathbf{x}, \mathbf{u}) = f(\mathbf{x}) + \sum_{j=1}^{l} u_j g_j(\mathbf{x}), \tag{7—15}$$

în care $u_1, u_2, \ldots, u_l$ sînt *multiplicatorii Lagrange*.

Condiţia necesară de minim a funcţiei, (7—15), constă în anularea derivatelor parţiale de ordinul întîi:

$$\frac{\partial F}{\partial x_i} = \frac{\partial f}{\partial x_i} + \sum_{j=1}^{l} u_j \frac{\partial g_j}{\partial x_i} = 0, \ (i = 1, 2, \ldots m); \tag{7—16}$$

$$\frac{\partial F}{\partial u_j} = g_j = 0, \ (j = 1, 2, \ldots, l). \tag{7—17}$$

Relaţiile (7—16) şi (7—17) conţin $m + l$ ecuaţii, fapt ce permite determinarea celor $m + l$ necunoscute. Aceste relaţii reprezintă numai condiţiile necesare de minim; ele sînt şi suficiente numai în anumite cazuri particulare.

Ex. 7—1. Să se rezolve problema [5]:

$$min \{x_1^2 + x_2^2 \mid x_1^2 + 4x_2^2 - 4x_1 - 12 = 0\}. \tag{7—18}$$

Lagrangeanul acestei probleme este:

$$F(x_1, x_2, u) = x_1^2 + x_2^2 + u(x_1^2 + 4x_2^2 - 4x_1 - 12) =$$
$$= x_1^2 + x_2^2 + u[(x_1 - 2)^2 + 4x_2^2 - 16]. \tag{7—19}$$

Scanned with OKEN Scanner

Derivatele parţiale ale funcţiei F în raport cu x_1, x_2 şi u conduc la sistemul:

$$\begin{cases} 2x_1 + 2(x_1 - 2)u = 0; \\ 2x_2 + 8x_2u = 0; \\ (x_1 - 2)^2 + 4x_2^2 - 16 = 0. \end{cases} \qquad (7-20)$$

Sistemul (7—20) are patru soluţii. În tabela 7—1 sînt prezentate atît valorile acestor soluţii, cît şi valoarea funcţiei de eficienţă pentru fiecare caz în parte.

Din calculul direct al valorii funcţiei de eficienţă se observă că punctele C şi D reprezintă minimuri absolute, punctul B — maxim relativ, iar punctul A — maxim absolut (fig. 7.4).

Tabela 7—1

Soluţiile problemei (7—18)

	A	B	C	D
x_1	6	−2	−2/3	−2/3
x_2	0	0	2 5/3	−2 5/3
u	−3/2	−1/2	−1/4	−1/4
f	36	4	24/9	24/9

Dacă funcţia de eficienţă f este convexă, iar restricţiile $\mathbf{g}(\mathbf{x})$ sînt liniare, se demonstrează că relaţiile (7—16) şi (7—17) sînt şi suficiente. În acest sens, prezentăm exemplul următor.

Ex. 7—2. Să se rezolve problema:

$$min\{x_1^2 + 0{,}75x_2^2 + 0{,}444x_3^2 + 0{,}187x_4^2 | x_1 + x_2 + x_3 + x_4 - 10 = 0\}. \qquad (7-21)$$

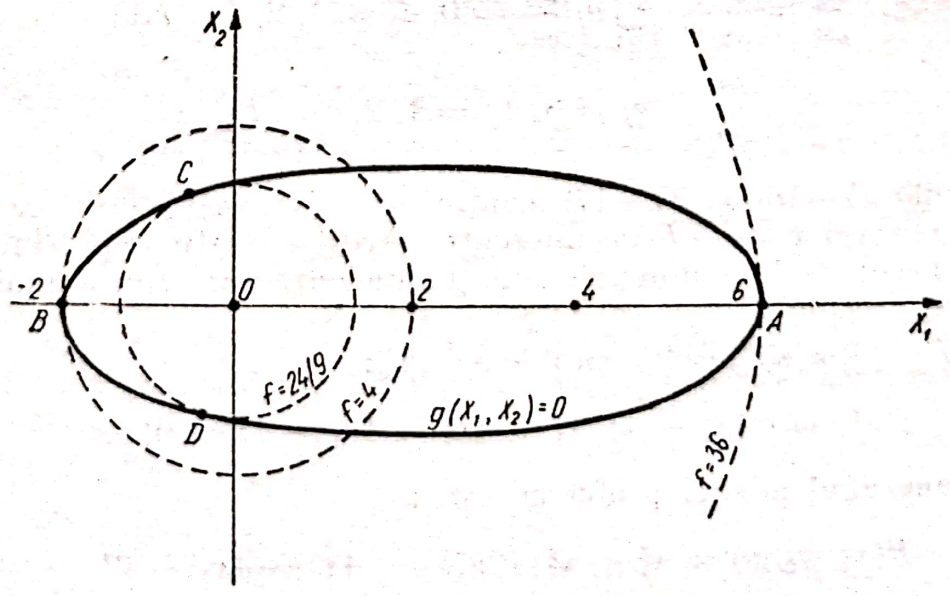


Fig. 7.4

Scanned with OKEN Scanner

Din enunț se observă că funcția de eficiență, fiind o sumă de pătrate, este convexă, iar restricția (legătura) este liniară.

Utilizînd multiplicatorul Lagrange u, se alcătuiește funcția asociată:

$$F(x_1, x_2, x_3, x_4, u) = x_1^2 + 0{,}75x_2^2 + 0{,}444x_3^2 + 0{,}187x_4^2 + \\ + u(x_1 + x_2 + x_3 + x_4 - 10), \qquad (7-22)$$

pentru care aplicarea relațiilor (7—16) și (7—17) conduce la ecuațiile:

$$\begin{cases} 2x_1 + u = 0; \\ 1{,}5x_2 + u = 0; \\ 0{,}888x_3 + u = 0; \\ 0{,}374x_4 + u = 0; \\ x_1 + x_2 + x_3 + x_4 = 10. \end{cases} \qquad (7-23)$$

Soluția sistemului (7—23) este:

$$x_1 = 1{,}006728; \quad x_2 = 1{,}342305; \quad x_3 = 2{,}2674075; \quad x_4 = 5{,}383577;$$

$$u = -2{,}0134579,$$

pentru care funcția de eficiență are valoarea minimă

$$f = 1{,}005^2 + 0{,}75 \times 1{,}34^2 + 0{,}444 \times 2{,}265^2 + 0{,}187 \times 5{,}39^2 = 4{,}03.$$

Condițiile Kuhn-Tucker. Atunci cînd problema de programare neliniară se prezintă sub forma generală (7—10), restricțiile fiind de tip inegalitate, problemei i se asociază l variabile auxiliare w_j. În acest mod, restricțiile:

$$g_j(\mathbf{x}) = g_j(x_1, x_2, \ldots, x_m) \leqslant 0, \quad (j = 1, 2, \ldots, l), \qquad (7-24)$$

prin adăugarea variabilelor auxiliare w_j, se transformă în egalitățile:

$$g_j(\mathbf{x}) = w_j^2 = 0, \quad (j = 1, 2, \ldots, l). \qquad (7-25)$$

Cu aceasta, problema de minim se poate exprima sintetic sub forma:

$$min\,\{f(\mathbf{x}) \mid g_j(\mathbf{x}) + w_j^2 = 0\}. \qquad (7-26)$$

Funcția lui Lagrange corespunzătoare acestei probleme de minim este:

$$F(\mathbf{x}, \mathbf{w}, \mathbf{u}) = f(\mathbf{x}) + \sum_{j=1}^{l} u_j [g_j(\mathbf{x}) + w_j^2]. \qquad (7-27)$$

În acest caz, condițiile necesare de minim sînt:

$$\frac{\partial F}{\partial x_i} = \frac{\partial f}{\partial x_i} + \sum_{j=1}^{l} u_j \frac{\partial g_j}{\partial x_i} = 0, \quad (i = 1, 2, \ldots, m); \qquad (7-28)$$

$$\frac{\partial F}{\partial u_j} = g_j + w_j^2 = 0, \quad (j = 1, 2, \ldots, l); \qquad (7-29)$$

$$\frac{\partial F}{\partial w_j} = 2u_j w_j = 0, \quad (j = 1, 2, \ldots, l). \qquad (7-30)$$

Scanned with OKEN Scanner

Problema (7—26), (7—27) conține $2l + m$ necunoscute; ecuațiile (7—28), ..., (7 — 30), în număr de $2l + m$, permit determinarea acestora.

Eliminarea variabilelor auxiliare w_j din relațiile (7—28), ... (7—30) conduce la rezultatul:

$$\frac{\partial f}{\partial x_i} + \sum_{j=1}^{l} u_j \frac{\partial g_j}{\partial x_i} = 0, \quad (i = 1, 2, \ldots, m); \tag{7—31}$$

$$g_j \leqslant 0, \quad (j = 1, 2, \ldots, l); \tag{7—32}$$

$$u_j g_j = 0, \quad (j = 1, 2, \ldots, l). \tag{7—33}$$

Ecuațiile (7—31), ..., (7—33) reprezintă doar condiții necesare de minim pentru problema (7—26). K u h n și T u c k e r, care au extins metoda multiplicatorilor lui Lagrange la probleme de optim cu restricții sub formă de inegalități, au adăugat în plus condiția:

$$u_j \geqslant 0, \quad (j = 1, 2, \ldots, l), \tag{7—34}$$

obligatorie pentru ca funcția f să aibă un minim în punctul respectiv.

Condițiile de optim (7—31), ..., (7—34), cunoscute sub numele de *condițiile Kuhn-Tucker*, sînt și *suficiente* atunci cînd funcțiile f și g_j sînt convexe.

7.2. ALGORITMI PENTRU PROGRAMELE NELINIARE

Există o varietate foarte mare de algoritmi pentru soluționarea problemelor de programare neliniară. O clasificare, și de aici o apreciere asupra eficacității acestora, este dificilă. Totuși, este util să se cunoască faptul că există o grupă de *algoritmi generali*, utilizabili la rezolvarea oricărei probleme de programare neliniară (convexă) și o altă grupă de algoritmi în care sînt incluse *metode dezvoltate pentru anumite genuri de probleme* de programare neliniară. O clasificare interesantă, alcătuită din acest punct de vedere, este efectuată în lucrarea [19]. Din punctul de vedere al utilizatorului de algoritmi de optimizare, această clasificare este printre cele mai utile.

În categoria algoritmilor generali sînt incluși:

— algoritmi bazați pe liniarizarea secțională;
— algoritmi tip gradient;
— algoritmul simplex;
— algoritmul explorării;
— algoritmi bazați pe căutarea întîmplătoare etc.

Din cea de a II-a categorie fac parte o serie de algoritmi cu ajutorul cărora se rezolvă unele cazuri speciale de programare neliniară, cum sînt: programarea pătratică, programarea segmentar-liniară, programarea monotonă, minimizarea sumei modulelor funcțiilor liniare etc.

Algoritmul liniarizării secționale. Fie domeniul $\mathbf{D}_N \subset R^m$ și funcția $f: \mathbf{D}_N \to R$, derivabilă. Fie, de asemenea, funcțiile $g_j: R^m \to R$, derivabile, și restricțiile:

$$P_{1j} \leqslant g_j(\mathbf{x}) \leqslant P_{2j}, \quad (j = 1, 2, \ldots, l) \tag{7—35}$$

unde $P_{1j} \in R$, $P_{2j} \in R$ și $\mathbf{x} \in \mathbf{X}$, $\mathbf{X}$ fiind o mulțime convexă.

Scanned with OKEN Scanner

Se formulează problema de optim:

$$max\,\{f(\mathbf{x}) \mid \mathbf{P}_1 \leqslant \mathbf{g}(\mathbf{x}) \leqslant \mathbf{P}_2,\ \mathbf{x} \geqslant 0\}. \tag{7—36}$$

Domeniul $\mathbf{D}_N$ introdus este definit de restricțiile:

$$\mathbf{P}_1 \leqslant \mathbf{g}(\mathbf{x}) \leqslant \mathbf{P}_2,\ \mathbf{x} \geqslant 0.$$

Dacă funcția $f(\mathbf{x})$ este concavă, va exista un singur punct de maxim; în caz contrar rezultă mai multe maximuri locale, dintre ele urmînd a se determina maximul absolut. În cele ce urmează se consideră că $f(\mathbf{x})$ este concavă.

Prin liniarizare se asociază problemei neliniare (7—36) problema liniară:

$$\begin{gathered} f_L(\mathbf{x}) = f(\mathbf{x}_0) + \Delta f(\mathbf{x}); \\ P_{1j} \leqslant g_j(\mathbf{x}_0) + \Delta g_j(\mathbf{x}) \leqslant P_{2j},\ (j = 1, 2, \ldots, l); \\ x_i \geqslant 0 \quad (i = 1, 2, \ldots, m). \end{gathered} \tag{7—37}$$

Pentru funcția liniară $f_L(\mathbf{x})$ avem:

$$max\, f_L(\mathbf{x}) = max[f_L(\mathbf{x}_0) + \Delta f(\mathbf{x})] = f_L(\mathbf{x}_0) + max\, \Delta f(\mathbf{x}). \tag{7—38}$$

Cu înlocuirea $\mathbf{x} - \mathbf{x}_0 = \Delta\mathbf{x}$ și ținînd seama de (7—37) și (7—38) rezultă:

$$\begin{gathered} f(\mathbf{x}_0) + max\, \Delta f(\Delta\mathbf{x}); \\ P_{1j} - g_j(\mathbf{x}_0) \leqslant \Delta g_j(\Delta x) \leqslant P_{2j} - g_j(\mathbf{x}_0),\ (j = 1, 2, \ldots, l); \\ -\Delta\mathbf{x}_L \leqslant \Delta\mathbf{x} \leqslant \Delta\mathbf{x}_L; \\ \mathbf{x}_0 + \Delta\mathbf{x} \geqslant 0; \end{gathered} \tag{7—39}$$

sau

$$\begin{gathered} max\, \Delta f(\Delta\mathbf{x}); \\ P_{1j} - g_j(\mathbf{x}_0) \leqslant \Delta g_j(\Delta\mathbf{x}) \leqslant P_{2j} - g_j(\mathbf{x}_0),\quad (j = 1, 2, \ldots, l); \\ -\Delta\mathbf{x}_L \leqslant \Delta\mathbf{x} \leqslant \Delta\mathbf{x}_L; \\ \mathbf{x}_0 + \Delta\mathbf{x} \geqslant 0, \end{gathered} \tag{7—40}$$

unde $\pm\Delta\mathbf{x}_L$ reprezintă limitele superioară, respectiv inferioară ale lui $\Delta\mathbf{x}$.

Restricțiile:

$$\begin{gathered} P_{1j} - g_j(\mathbf{x}_0) \leqslant g_j(\Delta\mathbf{x}) \leqslant P_{2j} - g_j(\mathbf{x}_0),\ (j = 1, 2, \ldots, l); \\ -\Delta x_{Li} \leqslant \Delta x_i \leqslant \Delta x_{Li}; \\ x_{0i} + \Delta x_i \geqslant 0,\ (i = 1, 2, \ldots, m), \end{gathered} \tag{7—41}$$

definesc domeniul convex $\mathbf{D}_L \subset \mathbf{D}_N$.

Formularea (7—40) reprezintă o problemă de programare liniară pe domeniul $\mathbf{D}_L \subset \mathbf{D}_N$. Soluționarea acesteia conduce la rezultatul:

$$\Delta\hat{\mathbf{x}},\ \Delta f(\Delta\hat{\mathbf{x}}) \text{ și } \Delta\mathbf{g}(\Delta\hat{\mathbf{x}}). \tag{7—42}$$

Scanned with OKEN Scanner

Cu soluţia (7—42) se obţine:

$$\hat{\mathbf{x}}_1 = \mathbf{x}_0 + \Delta\hat{\mathbf{x}};$$
$$\mathbf{g}_L(\hat{\mathbf{x}}_1) = \mathbf{g}(\mathbf{x}_0) + \Delta\mathbf{g}(\Delta\hat{\mathbf{x}}); \qquad (7-43)$$
$$f_L(\hat{\mathbf{x}}_1) = f(\mathbf{x}_0) + \Delta f(\Delta\hat{\mathbf{x}}) = f_{L1}.$$

Cu soluţia $\hat{\mathbf{x}}_1$ din (7—43) se determină valoarea funcţiei de eficienţă neliniare f:

$$f(\hat{\mathbf{x}}_1) = f_1. \qquad (7-44)$$

Dacă $f_1 > f(\mathbf{x}_0) = f_0$, atunci problema de programare neliniară (7—36) se liniarizează din nou în punctul $\hat{\mathbf{x}}_1$ şi, rezolvînd acum problema de programare liniară (7—40) în punctul $\hat{\mathbf{x}}_1 \in \mathbf{D}_L$, cu ajutorul relaţiilor (7—43) se obţine:

$$\hat{\mathbf{x}}_2, f_{L2} \text{ şi } f_2.$$

Dacă $f_2 > f_1$, problema neliniară se liniarizează în noul punct $\hat{\mathbf{x}}_2$, se rezolvă problema de programare liniară asociată şi se obţine în final:

$$\hat{\mathbf{x}}_3, f_{L3} \text{ şi } f_3.$$

Continuînd în acest mod se ajunge, la un moment dat, la situaţia:

$$f_{k+1} < f_k, \qquad (7-45)$$

ceea ce arată că s-a depăşit maximul funcţiei concave $f(\mathbf{x})$ în domeniul său de definiţie $\mathbf{D}_N$. În această situaţie se execută o reîntoarcere la punctul anterior:

$$\hat{\mathbf{x}}_k := \hat{\mathbf{x}}_{k-1} - \Delta\hat{\mathbf{x}}, \qquad (7-46)$$

se finisează pasul prin divizarea lui $\Delta\mathbf{x}$ cu numărul n_d:

$$\Delta\mathbf{x} := \Delta\mathbf{x}/n_d \qquad (7-47)$$

şi se continuă operaţiile de mai sus.

Operaţiile de *întoarcere* şi *finisare a pasului* continuă pînă cînd se atinge pentru $\Delta\mathbf{x}$ o limită inferioară, aprioric fixată. În final se obţine şirul:

$$f_0, f_1, f_2, \ldots, f_k, \ldots, f_n, \qquad (7-48)$$

care este monoton crescător şi mărginit.

Caracteristica de şir monoton crescător rezultă din modul în care a fost generat, iar mărginirea se datoreşte faptului că maximul funcţiei $f(\mathbf{x})$ pe domeniul $\mathbf{D}_N$ este finit.

Fie $M = max\{f(\mathbf{x}) \mid \mathbf{x} \subset \mathbf{D}_N\}$. În acest caz, atunci cînd $\Delta\mathbf{x}$ a atins limita inferioară ε, aprioric fixată, avem:

$$f_n - M < \eta(\varepsilon), \qquad (7-49)$$

unde ε este un „infinit mic".

Scanned with OKEN Scanner

În momentul în care condiția (7—49) a fost îndeplinită, căutarea optimului încetează, el fiind atins cu precizia stabilită anticipat prin valoarea lui ε.

După cum se observă, determinarea optimului problemei neliniare se face pe secțiuni, pe domenii $\mathbf{D}_L$, unde problemei neliniare i se asociază o alta liniară. De aici provine denumirea de *programare liniară secțională* a acestui mod de abordare a problemei.

Pentru a fi mai ușor programată, problema (7—40) se prelucrează în continuare. Astfel, prin retranscrierea restricțiilor cu limitarea într-o singură parte se obține:

$$\begin{aligned} &max\Delta f(\Delta \mathbf{x}); \\ &\Delta g_j(\Delta \mathbf{x}) \geqslant P_{1j} - g_j(\mathbf{x}_0); \\ &\Delta g_j(\Delta \mathbf{x}) \leqslant P_{2j} - g_j(\mathbf{x}_0) \quad (j = 1, 2, \ldots, l); \\ &\Delta x \geqslant -\Delta \mathbf{x}_L; \\ &\Delta \mathbf{x} \leqslant \Delta \mathbf{x}_L; \\ &\mathbf{x}_0 + \Delta \mathbf{x} \geqslant 0. \end{aligned} \tag{7—50}$$

În continuare se face următoarea schimbare de variabilă:

$$\Delta x_i = \Delta x_i^* - \Delta x_{Li} \quad (i = 1, 2, \ldots, m), \tag{7—51}$$

necesară pentru îndeplinirea condițiilor de pozitivitate ale variabilelor la programarea liniară.

Înlocuind (7—51) în (7—50) rezultă:

$$\begin{aligned} &max\Delta f(\Delta \mathbf{x}^* - \Delta \mathbf{x}_L); \\ &\Delta g_j(\Delta \mathbf{x}^* - \Delta \mathbf{x}_L) \geqslant P_{1j} - g_j(\mathbf{x}_0); \\ &\Delta g_j(\Delta \mathbf{x}^* - \Delta \mathbf{x}_L) \leqslant P_{2j} - g_j(\mathbf{x}_0) \quad (j = 1, 2, \ldots, l); \\ &\Delta \mathbf{x}^* \geqslant 0; \\ &\Delta \mathbf{x}^* \leqslant 2\Delta \mathbf{x}_L; \\ &\Delta \mathbf{x}^* \geqslant \Delta \mathbf{x}_L - \mathbf{x}_0. \end{aligned} \tag{7—52}$$

Deoarece funcțiile $\Delta f(.)$ și $g(.)$ sînt liniare, rezultă:

$$\begin{aligned} &-\Delta f(\Delta \mathbf{x}_L) + max\Delta f(\Delta \mathbf{x})^*; \\ &\Delta g_j(\Delta \mathbf{x}^*) \geqslant P_{1j} - g_j(\mathbf{x}_0) + \Delta g_j(\Delta \mathbf{x}_L); \\ &\Delta g_j(\Delta \mathbf{x}^*) \leqslant P_{2j} - g_j(\mathbf{x}_0) + \Delta g_j(\Delta \mathbf{x}_L) \quad (j = 1, 2, \ldots, l) \\ &\Delta x^* \leqslant 2\Delta \mathbf{x}_L; \\ &\Delta \mathbf{x}^* \geqslant \Delta \mathbf{x}_L - \mathbf{x}_0; \\ &\Delta \mathbf{x}^* \geqslant 0. \end{aligned} \tag{7—53}$$

Scanned with OKEN Scanner

Introducînd notaţiile:

$$b_{1j} = P_{1j} - g_j(\mathbf{x}_0) + \Delta g_j(\Delta \mathbf{x}_L);$$
$$b_{2j} = P_{2j} - g_j(\mathbf{x}_0) + \Delta g_j(\Delta \mathbf{x}_L) \quad (j = 1, 2, \ldots, l); \tag{7—54}$$
$$\mathbf{b}_3 = \Delta x_L - x_0;$$
$$\mathbf{b}_4 = 2\Delta x_L,$$

se obţine o problemă de programare liniară:

$$\begin{aligned} &max\ \Delta f(\Delta \mathbf{x}^*);\\ &\Delta g_j(\Delta \mathbf{x}^*) \geqslant b_{1j};\\ &\Delta g_j(\Delta \mathbf{x}^*) \leqslant b_{2j} \quad (j = 1, 2, \ldots, l);\\ &\Delta \mathbf{x}^* \geqslant \mathbf{b}_3;\\ &\Delta \mathbf{x}^* \leqslant \mathbf{b}_4;\\ &\Delta \mathbf{x}^* \geqslant 0. \end{aligned} \tag{7—55}$$

Rezolvînd problema (7—55) se obţine soluţia:

$$\Delta \hat{\mathbf{x}}^*,\ \Delta g_j(\Delta \hat{\mathbf{x}}^*) \quad \text{şi} \quad \Delta f(\Delta \hat{\mathbf{x}}^*). \tag{7—56}$$

Ţinînd seama de acest rezultat, relaţiile (7—43) devin:

$$\begin{aligned} \hat{\mathbf{x}}_1 &= \mathbf{x}_0 - \Delta \mathbf{x}_L + \Delta \hat{\mathbf{x}}^*;\\ \mathbf{g}_L(\hat{\mathbf{x}}_1) &= \mathbf{g}(\mathbf{x}_0) - \Delta \mathbf{g}(\Delta \mathbf{x}_L) + \Delta \mathbf{g}(\Delta \hat{\mathbf{x}}^*);\\ f_L(\hat{\mathbf{x}}_1) &= f(\mathbf{x}_0) - \Delta f(\Delta \mathbf{x}_L) + \Delta f(\Delta \hat{\mathbf{x}}^*). \end{aligned} \tag{7—57}$$

Observaţie. Sînt cazuri practice la care restricţia $\mathbf{x} \geqslant 0$ este înlocuită cu restricţia $\mathbf{x}_m \leqslant \mathbf{x} \leqslant \mathbf{x}_M$, unde $\mathbf{x}_m$ şi $\mathbf{x}_M$ reprezintă vectorii valorilor minime, respectiv maxime ale lui $\mathbf{x}$. De exemplu, pentru multe procese chimice, temperatura T este limitată atît inferior (T_m) cît şi superior (T_M). Într-un asemenea caz, restricţia de mai sus este de forma $T_m \leqslant T \leqslant T_M$.

Ex. 7—3. Să se stabilească formele concrete (7—55) şi (7—57) pentru problema:

$$\begin{aligned} max\ f(x_1, x_2) &= max(c - (x_1 - a)^2 - (x_2 - b)^2);\\ P_{11} \leqslant g_1(x_1, x_2) &= x_1 + a_{12}x_2^2 \leqslant P_{21};\\ P_{12} \leqslant g_2(x_1, x_2) &= a_{21}x_1 + x_2 \leqslant P_{22};\\ x_1 \geqslant 0;\quad x_2 &\geqslant 0, \end{aligned} \tag{7—58}$$

unde constantele c, a, b, a_{12}, a_{21}, P_{11}, P_{12} şi P_{22} sînt strict pozitive.

Funcţia $f(x_1, x_2) = c - (x_1 - a)^2 - (x_2 - b)^2$ este concavă în cadranul $x_1 \geqslant 0$, $x_2 \geqslant 0$. Pentru $x_1 = a$ şi $x_2 = b$, funcţia $f(x_1, x_2)$ ia valoarea maximă $f_0 = c$, care reprezintă optimul în lipsa restricţiilor g_1 şi g_2 (fig. 7.5). În prezenţa restricţiilor se obţine un punct de maxim cu o valoare a funcţiei de eficienţă f mai mică. Fie această valoare f_B, asociată punctului B.

Scanned with OKEN Scanner

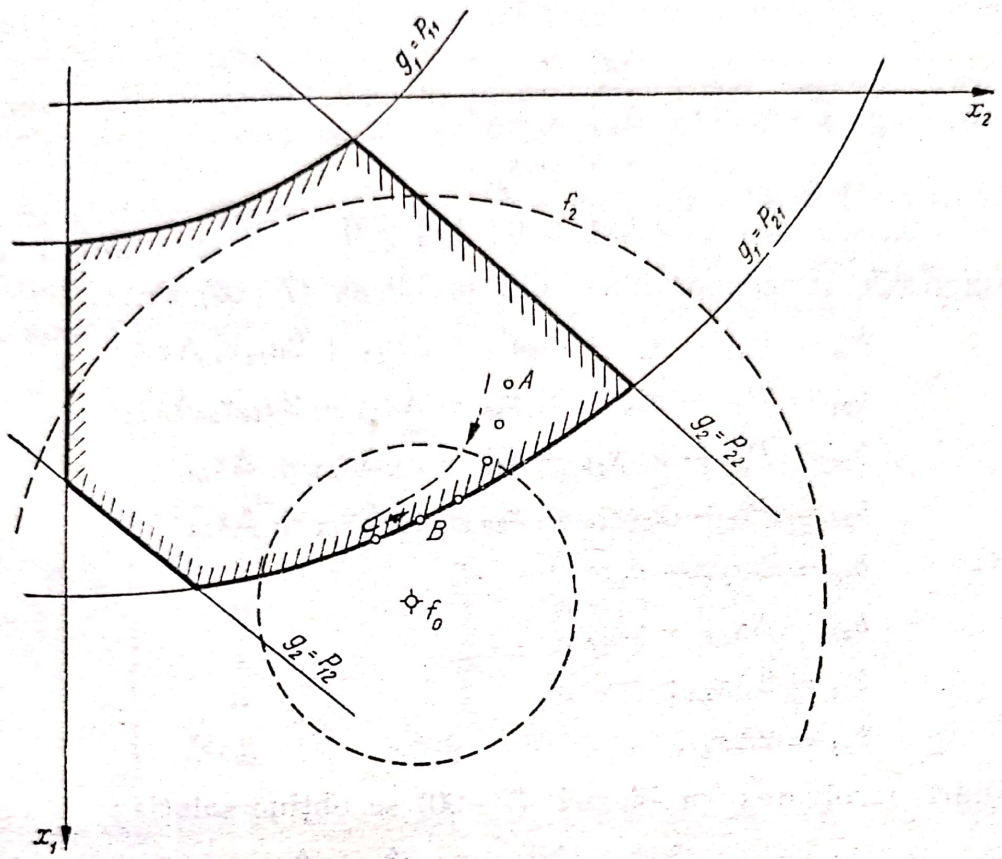


Fig. 7.5

Liniarizarea problemei (7—58) conduce la rezultatul:

$$max\ f_L(\Delta x_1, \Delta x_2) = max(c - (x_{10} - a)^2 - (x_{20} - b)^2 - 2(x_{10} - a)\Delta x_1 - \\ -2(x_{20} - b)\Delta x_2);$$

$$P_{11} \leqslant x_{10} + a_{12}x_{20}^2 + \Delta x_1 + 2a_{12}x_{20}\Delta x_2 \leqslant P_{21};$$

$$P_{12} \leqslant a_{21}x_{10} + x_{20} + a_{21}\Delta x_1 + \Delta x_2 \leqslant P_{22}; \qquad (7—59)$$

$$-\Delta x_{1L} \leqslant \Delta x_1 \leqslant \Delta x_{L1};$$

$$-\Delta x_{2L} \leqslant \Delta x_2 \leqslant \Delta x_{L2};$$

$$x_{10} + \Delta x_1 \geqslant 0;$$

$$x_{20} + \Delta x_2 \geqslant 0.$$

Punînd problema (7—59) sub forma (7—55) se obține:

$$max\ (-2(x_{10} - a)\Delta x_1^* - 2(x_{20} - b)\Delta x_2^*);$$

$$\Delta x_1^* + 2a_{12}x_{20}\Delta x_2^* \geqslant b_{11};$$

$$\Delta x_1^* + 2a_{12}x_{20}\Delta x_2^* \leqslant b_{12}; \qquad (7—60)$$

$$a_{21}\Delta x_1^* + \Delta x_2^* \geqslant b_{12};$$

$$a_{21}\Delta x_1^* + \Delta x_2^* \leqslant b_{22};$$

Scanned with OKEN Scanner

$$\Delta x_1^* \geqslant b_{13};$$
$$\Delta x_2^* \geqslant b_{23};$$
$$\Delta x_1^* \leqslant b_{14};$$
$$\Delta x_2^* \leqslant b_{24};$$
$$\Delta x_1^* \geqslant 0;\ \Delta x_2^* \geqslant 0.$$

Expresiile termenilor liberi din problema (7—60) sînt:

$$\begin{aligned}
b_{11} &= P_{11} - x_{10} - a_{12}a_{20}^2 + \Delta x_{L1} + 2a_{12}x_{20}\Delta x_{L2};\\
b_{21} &= P_{21} - x_{10} - a_{12}x_{20} + \Delta x_{L1} + 2a_{12}x_{20}\Delta x_{L2};\\
b_{12} &= P_{12} - a_{21}x_{10} - x_{20} + a_{21}\Delta x_{L1} + \Delta x_{L2};\\
b_{22} &= P_{22} - a_{21}x_{10} - x_{20} + a_{21}\Delta x_{L1} + \Delta x_{L2};\\
b_{13} &= \Delta x_{L1} - x_{10};\\
b_{23} &= \Delta x_{L2} - x_{20};\\
b_{14} &= 2\Delta x_{L1};\\
b_{24} &= 2\Delta x_{L2}.
\end{aligned} \tag{7—61}$$

După rezolvarea problemei (7—60) se obţine soluţia;

$$\Delta\hat{x}_1^*;\quad \Delta\hat{x}_2^*;\quad \Delta f(\Delta\hat{x}_1^*, \Delta\hat{x}_2^*),$$

în funcţie de care se calculează:

$$\begin{aligned}
\hat{x}_1 &= x_{10} - \Delta x_{L1} + \Delta\hat{x}_1^*;\\
\hat{x}_2 &= x_{20} - \Delta x_{L2} + \Delta\hat{x}_2^*;
\end{aligned} \tag{7—62}$$

$$f_L(\hat{x}_1, \hat{x}_2) = c - (x_{10} - a)^2 - (x_{20} - b)^2 + 2(x_{10} - a)\Delta x_{L1} + 2(x_{20} - b)\Delta x_{L2} + \Delta f(\Delta\hat{x}_1^*, \Delta\hat{x}_2^*).$$

*
* *

După parcurgerea acestui exemplu, care ne-a apropiat de înţelegerea metodei liniarizării secţionale, continuăm prezentarea algoritmului.

După cum s-a văzut în capitolul 6, unei probleme de programare liniară i se asociază o matrice, care, prin transformări succesive, conduce la soluţia optimală. În acest sens, pentru problema (7—55) se poate scrie o asemenea matrice. Pentru mărirea gradului de înţelegere se va scrie această matrice pentru problema (7—60) din exemplul 7—3.

Mai întîi observăm însă că restricţiile problemei de programare liniară (7—60) sînt atît de tipul „$\leqslant$" cît şi de tipul „$\geqslant$". Deoarece subrutina de programare liniară utilizată admite numai restricţii de forma „$\leqslant$" şi egalităţi „$=$", termenul liber trebuind în acelaşi timp să fie pozitiv, rezultă că

Scanned with OKEN Scanner

restricţiile de tipul „$\geqslant$" trebuie transformate în egalităţi prin adăugarea unor variabile de ecart.

Atunci cînd termenul liber al unei restricţii este negativ, aceasta se înmulţeşte cu -1, ceea ce atrage după sine schimbarea tipului restricţiei.

După cum se observă din expresiile termenilor liberi (7—61), ultimii doi sînt întotdeauna strict pozitivi. Aşadar, schimbarea tipului restricţiei poate interveni numai la şase din cele opt restricţii ale problemei (7—60).

Cu aceste precizări, matricea care se asociază problemei (7—60) este:

$$\mathbf{C} = \begin{pmatrix} 1 & 2a_{12}x_{20} & 0\,0\,0\,0\,0\,0 & b_{11} \\ 1 & 2a_{12}x_{20} & 0\,0\,0\,0\,0\,0 & b_{21} \\ a_{21} & 1 & 0\,0\,0\,0\,0\,0 & b_{12} \\ a_{21} & 1 & 0\,0\,0\,0\,0\,0 & b_{22} \\ 1 & 0 & 0\,0\,0\,0\,0\,0 & b_{13} \\ 0 & 1 & 0\,0\,0\,0\,0\,0 & b_{23} \\ 1 & 0 & 0\,0\,0\,0\,0\,0 & b_{14} \\ 0 & 1 & 0\,0\,0\,0\,0\,0 & b_{24} \\ -2(x_{10}-a) & -2(x_{20}-b) & 0\,0\,0\,0\,0\,0\,0 & \end{pmatrix} \qquad (7-63)$$

Primele şase linii ale matricei **C** corespund primelor şase restricţii care, fie că sînt de la început de tipul „$\geqslant$", fie că devin pe parcurs de acest tip (datorită termenilor liberi care pot fi negativi) şi prin înmulţirea cu -1, restricţiile de tipul „$\leqslant$" devin de tipul „$\geqslant$". Din aceste considerente rezultă că numărul maxim posibil de restricţii de tipul „$\geqslant$" este egal cu şase. Cele şase coloane „umplute" cu zerouri permit transformarea restricţiilor de tipul „$\geqslant$" în restricţii de tipul „$\leqslant$" prin adăugarea variabilelor de ecart corespunzătoare (pe linia şi coloana corespunzătoare din matricea **C** apare coeficientul -1).

O nouă variantă a programării liniare secţionale [18]. Pentru rezolvarea problemei de programare liniară (7—60) trebuie tratată corespunzător matricea **C**. Pentru aceasta există două posibilităţi:

— se rezolvă problema de programare liniară operînd cu întreaga matrice **C**;

— se rezolvă problema de programare liniară pentru care se folosesc numai restricţiile domeniului de liniarizare D_L plus cele din domeniul mare D_N, neîndeplinite.

Problemele de optim practice conţin, de obicei, un număr mare de restricţii, care conduc la dimensiuni apreciabile pentru matricea **C**. În aceste situaţii, rezolvarea *problemei de programare liniară* (denumită în continuare problema PL) se face cu un consum de timp la calculator important.

Din contră, cea de a doua cale conduce la probleme de programare liniară cu un număr mic de restricţii şi de aici la consumuri mici de timp. În figura 7.6 sînt ilustrate grafic, pentru exemplul 7—3, diferite domenii de liniarizare D_L şi numărul de restricţii ale problemei PL asociate.

Scanned with OKEN Scanner

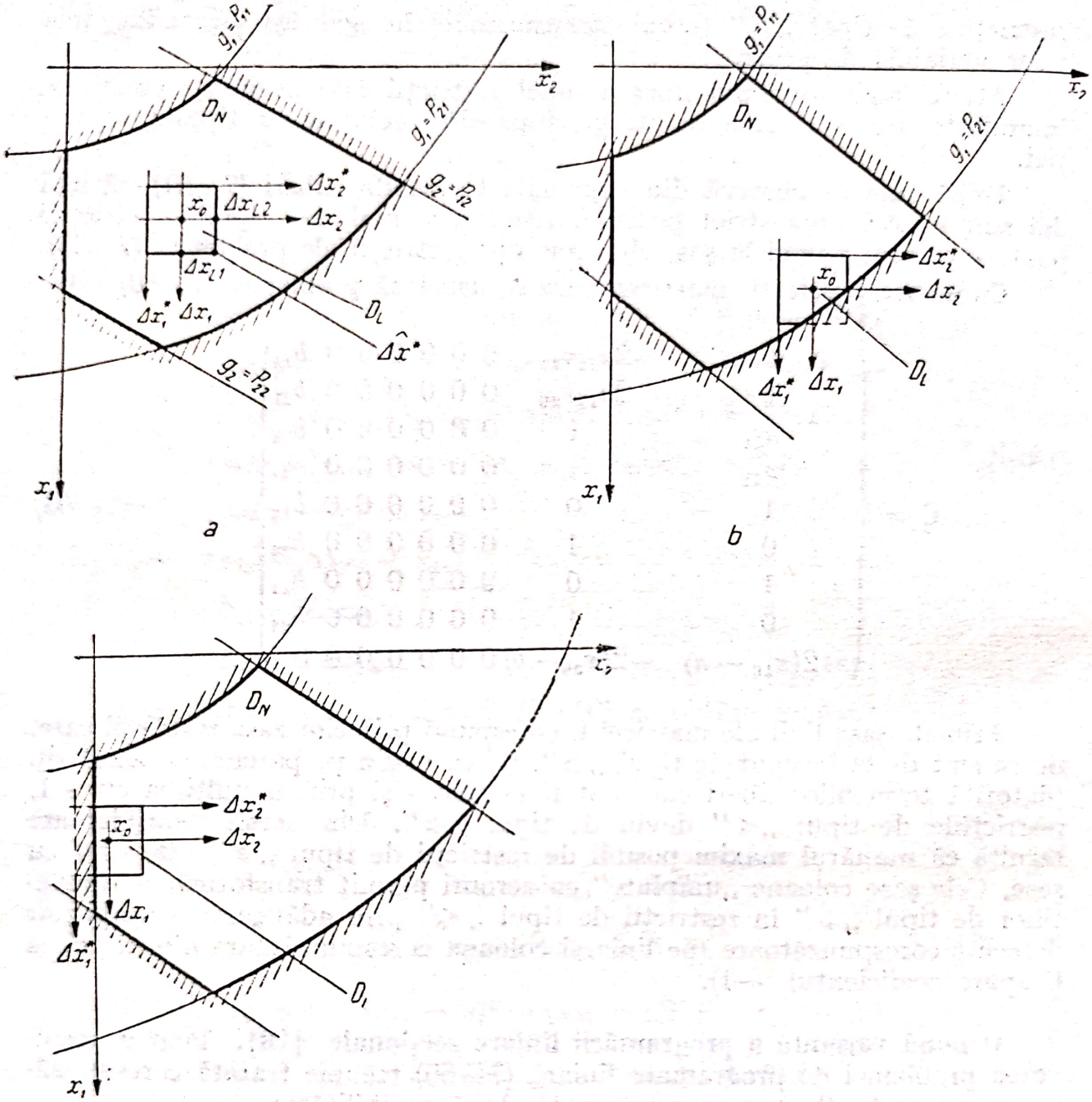


Fig. 7.6

Pentru situaţia din figura 7.6, *a*, punctul x_0 în care s-a efectuat liniarizarea este astfel situat încît restricţiile problemei PL sînt numai cele ale domeniului D_L:

$$\Delta x_1^* \leqslant b_{14}; \qquad (7-64)$$
$$\Delta x_2^* \leqslant b_{24}.$$

În acest caz, problema PL (7—60) ia forma foarte simplă:

$$max(-2(x_{10} - a)\Delta x_1^* - 2(x_{20} - b)\Delta x_2^*);$$
$$\Delta x_1 \leqslant b_{14}; \qquad (7-65)$$
$$\Delta x_2^* \leqslant b_{24}.$$

Scanned with OKEN Scanner

În cazul prezentat în figura 7.6, *b*, la restricţiile (7—64) se adaugă restricţia liniarizată :

$$\Delta x_1^* + a_{12}x_{20}\Delta x_2^* \leqslant b_{21},$$

asociată restricţiei neliniare a domeniului D_N :

$$x_1 + a_{12}x_2^2 \leqslant P_{21}.$$

Pentru această situaţie, problema PL (7—60) ia forma simplă :

$$\begin{gathered} max(-2(x_{10} - a)\Delta x_1^* - 2(x_{20} - b)\Delta x_2^*) ; \\ \Delta x_1^* + a_{12}x_{20}\Delta x_2^* \leqslant b_{21} ; \\ \Delta x_1^* \leqslant b_{14} ; \\ \Delta x_2^* \leqslant b_{24}. \end{gathered} \qquad (7—66)$$

În fine, pentru cazul reprezentat în figura 7.6, *c* trebuie inclusă restricţia $x_2 \geqslant 0$ asociată domeniului mare D_N şi care pentru problema liniarizată are forma $\Delta x_2^* \geqslant b_{23}$.

Astfel, problema PL ia forma particulară, de asemenea, foarte simplă :

$$\begin{gathered} max(-2(x_{10} - a)\Delta x_1^* - 2(x_{20} - b)\Delta x_2^*) ; \\ \Delta x_2^* \geqslant b_{23} ; \\ \Delta x_2^* \leqslant b_1 ; \\ \Delta x_2^* \leqslant b_{24}. \end{gathered} \qquad (7—67)$$

Din compararea formulării (7—60) cu cele de la (7—65), . . ., (7—67) se observă simplitatea acestora din urmă.

În figura 7.7 este prezentată, principial, schema de calcul simplificată a algoritmului programării liniare secţionale în varianta care utilizează pentru problema PL un număr mic de restricţii, aşa cum s-a văzut din cele prezentate mai sus.

Prin *nrgn* s-a notat numărul restricţiilor g_j neîndeplinite, valoarea zero a acestui indicativ marcînd situarea punctului x_0 în domeniul D_N. Cu $f_a(x_0)$ s-a notat valoarea iniţială, anterioară, a funcţiei de eficienţă.

La prima parcurgere a schemei de calcul avem $nrgn = 0$; se porneşte deci dintr-un punct aparţinînd domeniului D_N. După trecerea prin blocul de analiză a restricţiilor g_j, indicatorul *nrgn* rămîne mai departe cu valoarea zero. În continuare se generează matricea **A** (matricea simplex) şi vectorul U (marchează restricţiile tip egalitate), pentru subprogramul PL. Dacă valoarea funcţiei de eficienţă neliniare f în noul punct de optim x_0 este mai mare decît valoarea anterioară a acesteia, $f > f_a$ (se cercetează funcţii concave), se continuă determinarea optimului pentru un nou pas de liniarizare. În caz contrar, $f < f_a$, punctul de maxim a fost depăşit şi urmează o reîntoarcere la punctul anterior, finisarea pasului Δx_L prin împărţirea acestuia cu n_d, după care se reia liniarizarea şi determinarea soluţiei în condiţiile pasului micşorat.

Atunci cînd a fost depăşită graniţa domeniului problemei neliniare D_N, după parcurgerea blocului de analiză a restricţiilor va rezulta $nrgn \neq 0$.

Scanned with OKEN Scanner

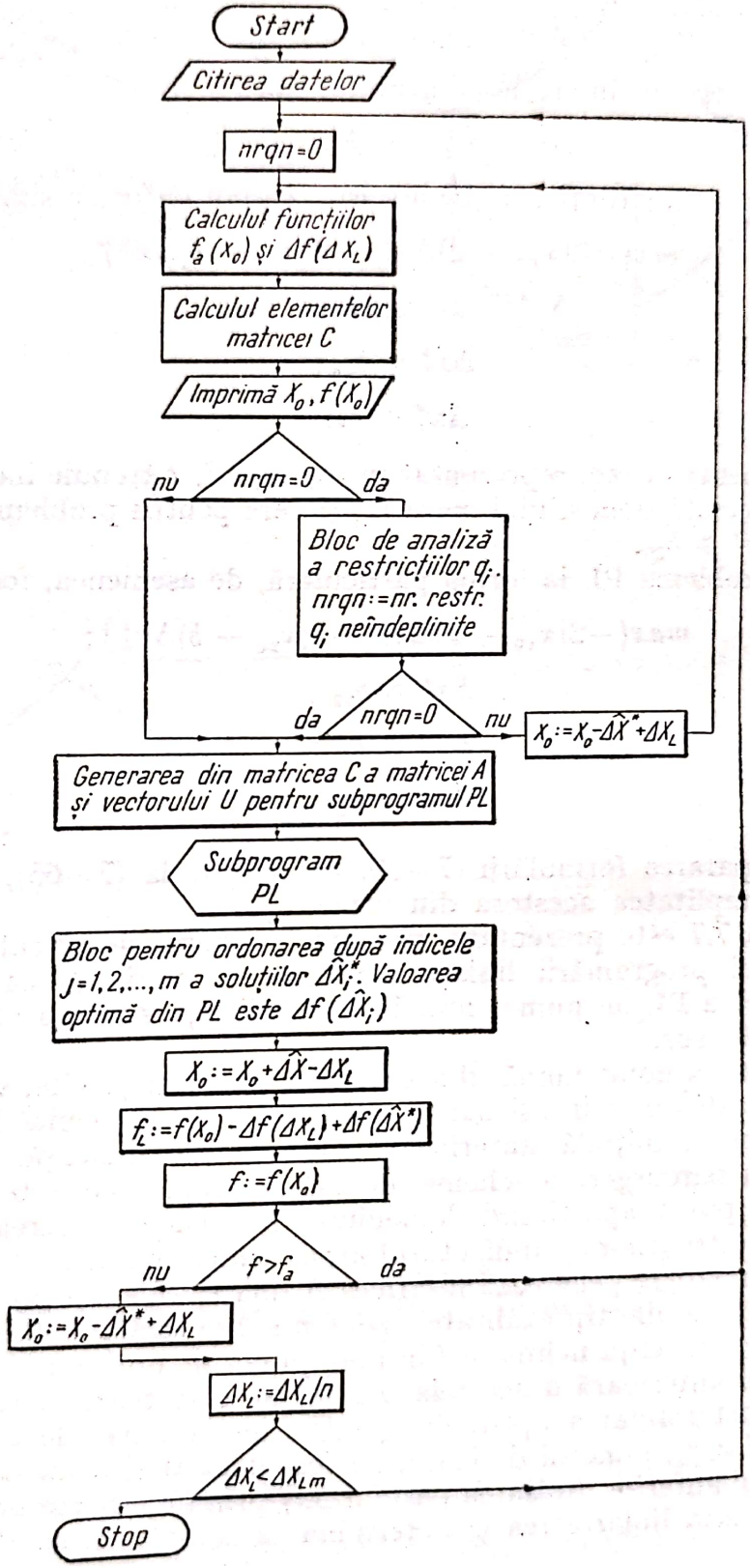


Fig. 7.7

Scanned with OKEN Scanner

În acest caz urmează o reîntoarcere la punctul x_0 anterior, $x_0 := x_0 - \Delta x^* + \Delta x_L$, includerea restricției neîndeplinite în modelul liniarizat și rezolvarea problemei de optim în aceste condiții (v. fig. 7.6, *b* și *c*).

Condiția de terminare a căutării optimului problemei neliniare este dată în schema de calcul prin atingerea unei valori minime, aprioric fixată, a pasului de liniarizare Δx_{Lm}.

Pentru exemplificare, într-un caz numeric concret, a fost rezolvată problema din *Ex. 7—3* cu următoarele date:

$c = 100$; $a = 8$; $b = 6$; $a_{12} = 0{,}035$; $a_{21} = -0{,}5$;

$P_{11} = 4$; $P_{21} = 8$; $P_{12} = -5$; $P_{22} = 6$;

punctul inițial: $x_{10} = 5$; $x_{20} = 8$;

valorile inițiale ale pașilor: $\Delta x_1 = 1$; $\Delta x_2 = 1$;

valorile minime admise pentru pași: $\Delta x_{Lm1} = 0{,}1$; $\Delta x_{Lm2} = 0{,}1$.

Calitativ, în figura 7.5, și cantitativ, în tabela 7—2, se pot urmări etapele parcurse din punctul inițial x_0 (5; 8) pînă la punctul corespunzător soluției optime x(6,95; 5,5), unde $f = 98{,}64749$.

Tabela 7—2

Nr. pas	Valorile variabilelor independente, restricțiilor și funcției de eficiență	Restricțiile utilizate pentru modelul liniarizat	Observații
1	2	3	4
1	$x_1 = 5$ $x_2 = 8$ $g_1 = 7{,}24$ $g_2 = 5{,}5$ $f = 87{,}0$	$\Delta x_1^* \leqslant 2$ $\Delta x_2^* \leqslant 2$	Punct inițial pentru procesul de stabilire a soluției optime
2	$x_1 = 6$ $x_2 = 7$ $g_1 = 7{,}714999$ $g_2 = 4{,}0$ $f = 95{,}0$	$\Delta x_1^* \leqslant 2$ $\Delta x_2^* \leqslant 2$	
3	$x_1 = 7$ $x_2 = 6$ $g_1 = 8{,}25999$ $g_2 = 2{,}5$ $f = 99{,}0$	$\Delta x_1^* \leqslant 2$ $\Delta x_2^* \leqslant 2$	În cursul acestui pas nu a fost respectată restricția $g_1 \leqslant P_{21}$($g_1 = 8{,}25999$ și $P_{21} = 8$). Urmează întoarcerea la punctul anterior și includerea restricției $g_1 \leqslant P_{21}$ și restricțiile modelului liniarizat.
4	$x_1 = 6{,}775002$ $x_2 = 6{,}0$ $g_1 = 8{,}035001$ $g_2 = 2{,}612499$ $f = 98{,}49937$	$\Delta x_1^* + 0{,}49\Delta x_2^* \leqslant 1{,}775$ $\Delta x_1^* \leqslant 2$ $\Delta x_2^* \leqslant 2$	În restricțiile modelului liniarizat a fost inclusă și restricția $\Delta g_1(\Delta x_1^*, \Delta x_2^*) \leqslant b_{11}$.

Scanned with OKEN Scanner

Nr. pas	Valorile variabilelor independente, restricţiilor şi funcţiei de eficienţă	Restricţiile utilizate pentru modelul liniarizat	Observaţii
1	2	3	4
5	$x_1 = 7{,}160001$ $x_2 = 5{,}0$ $g_1 = 8{,}035$ $g_2 = 1{,}42$ $f = 98{,}29439$	$\Delta x_1^* + 0{,}42\Delta x_2^* \leqslant 1{,}385$ $\Delta x_1^* \leqslant 2$ $\Delta x_2^* \leqslant 2$	Deoarece $f = 98{,}29439$ este mai mică decît $f = 98{,}49937$, rezultă că maximul a fost depăşit. Urmează reîntoarcerea la punctul anterior şi finisarea pasului ($\Delta x_1 = \Delta x_2 = 0{,}5$).
6	$x_1 = 6{,}95$ $x_2 = 5{,}5$ $g_1 = 8{,}00875$ $g_2 = 2{,}025$ $f = 98{,}64749$	$\Delta x_1^* + 0{,}42\Delta x_2 \leqslant 0{,}675$ $\Delta x_1^* \leqslant 1{,}0$ $\Delta x_2^* \leqslant 1{,}0$	$\Delta x_1 = 0{,}5$ $\Delta x_2 = 0{,}5$
7	$x_1 = 6{,}749751$ $x_2 = 6{,}0$ $g_1 = 8{,}008749$ $g_2 = 2{,}025001$ $f = 98{,}43437$	$\Delta x_1^* + 0{,}385\Delta x_2^* \leqslant 1{,}6175$ $\Delta x_1^* \leqslant 1{,}0$ $\Delta x_2^* \leqslant 1{,}0$	Deoarece noua valoare (98,43437) a funcţiei de eficienţă este mai mică decît valoarea anterioară (98,64749) urmează reîntoarcerea la punctul anterior şi finisarea pasului ($\Delta x_1 = \Delta x_2 = 0{,}25$).
8	$x_1 = 6{,}845001$ $x_2 = 5{,}750$ $g_1 = 8{,}008748$ $g_2 = 2{,}025001$ $f = 98{,}60347$	$\Delta x_1^* + 0{,}385\Delta x_2^* \leqslant 1{,}85875$ $\Delta x_1^* \leqslant 0{,}5$ $\Delta x_2^* \leqslant 0{,}5$	Deoarece şi această valoare ($f = 98{,}60347$) este mai mică decît valoarea de la care s-a plecat ($f = 98{,}64749$), urmează reîntoarcerea şi finisarea pasului ($\Delta x_1 = \Delta x_2 = 0{,}125$).
9	$x_1 = 6{,}893126$ $x_2 = 5{,}6250$ $g_1 = 8{,}002626$ $g_2 = 2{,}178437$ $f = 98{,}63420$	$\Delta x_1^* + 0{,}385\Delta x_2^* \leqslant 1{,}976$ $\Delta x_1^* \leqslant 0{,}25$ $\Delta x_2^* \leqslant 0{,}25$	Deoarece şi această valoare ($f = 98{,}63420$) este mai mică decît valoare de la început ($f = 98{,}64749$), urmează reîntoarcerea şi finisarea pasului ($\Delta x_1 = \Delta x_2 = 0{,}0625$). Calculul se întrerupe deoarece pasul finisat este mai mic decît limita anterior fixată, $\Delta x_{Lm1} = \Delta x_{Lm2} = 0{,}1$.

Atunci cînd funcţia de eficienţă prezintă mai multe maximuri, utilizarea algoritmului liniarizării secţionale nu conduce, în general, la maximul absolut. Pentru determinarea acestuia este necesară aplicarea algoritmului pornind din diferite puncte ale domeniului D_N, ceea ce poate conduce la diferite puncte de maxim.

Din compararea punctelor de maxim determinate se poate stabili maximul absolut şi deci soluţia problemei.

Scanned with OKEN Scanner

Algoritmi gradient. Există diferite variante de algoritmi bazați pe conceptul de gradient. În cele ce urmează va fi prezentată o variantă caracterizată prin facilități pentru tehnica numerică de calcul.

Pornind dintr-un punct oarecare A se execută un pas de o mărime dată în direcția de creștere maximă a funcției de eficiență, adică în direcția gradientului. Deplasările de acest gen sînt perpendiculare pe liniile de nivel (fig. 7.8).

Este important de văzut cum se determină gradientul pentru fiecare pas. Pasul dat Δs ne conduce la observația că următorul punct trebuie să se găsească pe un cerc în jurul punctului A, fapt pentru care se poate scrie:

$$\Delta x_1^2 + \Delta x_2^2 = \Delta s^2. \qquad (7-68)$$

Ca rezultat al efectuării pasului se modifică valoarea funcției de eficiență cu Δf. Pe de altă parte, se cunoaște că această variație este:

$$\Delta f = \frac{\partial f}{\partial x_1}\Delta x_1 + \frac{\partial f}{\partial x_2}\Delta x_2. \qquad (7-69)$$

Problema care se pune este de a determina pe Δx_1 și Δx_2 astfel încît variația Δf să fie maximă. Deoarece mărimile Δx_1 și Δx_2 sînt supuse restricției (7—68), rezultă că avem de soluționat o problemă de maxim cu legături. Pentru acest caz, lagrangeanul este (v. 7—15):

$$F(\Delta x_1, \Delta x_2, u) = \frac{\partial f}{\partial x_1}\Delta x_1 + \frac{\partial f}{\partial x_2}\Delta x_2 + u(\Delta x_1^2 + \Delta x_2^2 - \Delta s^2). \qquad (7-70)$$

Anulînd derivatele de ordinul întîi ale funcției F se obține sistemul:

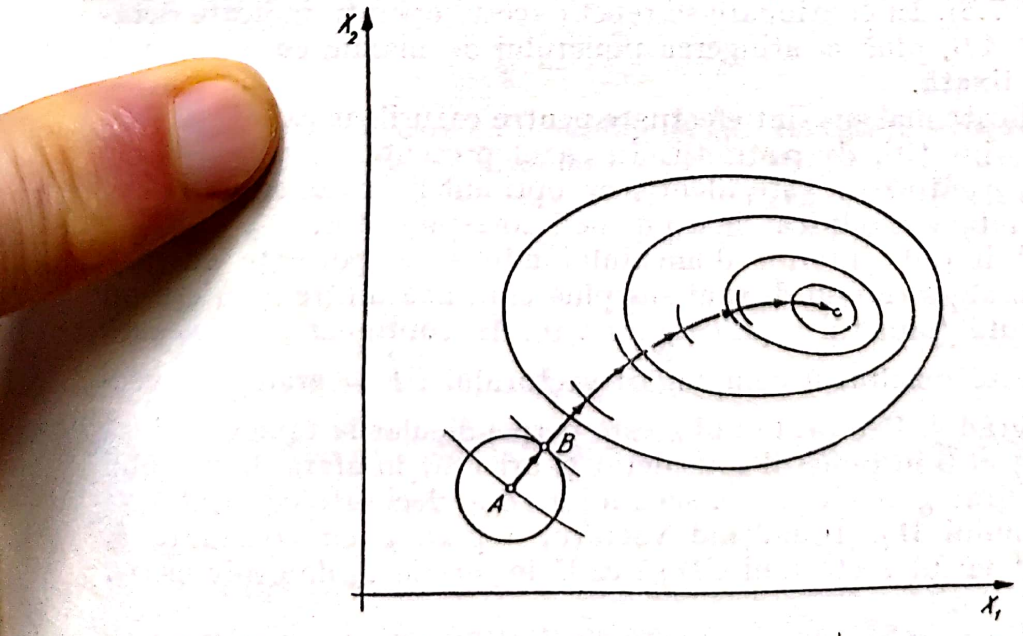


Fig. 7.8

Scanned with OKEN Scanner

$$\frac{\partial F}{\partial \Delta x_1} = \frac{\partial f}{\partial x_1} + 2u\Delta x_1 = 0;$$
$$\frac{\partial F}{\partial \Delta x_2} = \frac{\partial f}{\partial x_2} + 2u\Delta x_2 = 0; \qquad (7-71)$$
$$\frac{\partial F}{\partial u} = \Delta x_1^2 + \Delta x_2^2 - \Delta s^2 = 0,$$

care, rezolvat, conduce la soluţia:

$$\Delta x_i = \pm \frac{\Delta s}{\sqrt{\sum \left(\frac{\partial f}{\partial x_i}\right)^2}} \frac{\partial f}{\partial x_i}, \quad (i = 1, 2). \qquad (7-72)$$

Semnul lui Δx_i este pozitiv în probleme de maxim şi negativ în cele de minim.

De obicei derivatele parţiale $\partial f/\partial x_i$ se înlocuiesc cu raportul diferenţelor finite, după cum urmează:

$$\left(\frac{\partial f}{\partial x_1}\right)_{x_2} \cong \left(\frac{\Delta f}{\Delta x_1}\right)_{x_2};$$
$$\qquad (7-73)$$
$$\left(\frac{\partial f}{\partial x_2}\right)_{x_1} \cong \left(\frac{\Delta f}{\Delta x_2}\right)_{x_1}.$$

Pentru calculul expresiilor (7—73) se porneşte cu o anumită valoare fixată pentru Δx_i, fie aceasta a. Cu această valoare se calculează diferenţele:

$$(\Delta f)_{x_2} = f(x_1 + a, x_2) - f(x_1, x_2); \qquad (7-74)$$
$$(\Delta f)_{x_1} = f(x_1, x_2 + a) - f(x_1, x_2),$$

care, împărţite la a, conduc la determinarea valorilor expresiilor (7—73). Prin înlocuirea acestora în (7—72), unde de asemenea avem pentru Δs o valoare fixată, se obţin valorile variabilelor Δx_i, care permit determinarea punctului B (fig. 7.8). În continuare se repetă aceste operaţii indicate detaliat pentru pasul AB, pînă la atingerea punctului de maxim cu o anumită precizie aprioric fixată.

Calculele indicate mai sus sînt efectuate pentru cazurile în care problema de optim nu este însoţită de restricţii. În cazul prezenţei restricţiilor, iar optimul supus la legături nu este identic cu optimul liber, algoritmul prezentat mai sus trebuie modificat într-un mod corespunzător.

Din punctul iniţial, interior domeniului admis, se porneşte cu paşi efectuaţi conform algoritmului de mai sus pînă cînd una dintre restricţii nu mai este satisfăcută (punctul C din figura 7.9). În continuare, se execută pasul CD, care este rezultatul compunerii vectorului $\overrightarrow{CE} = \text{grad } f$ cu vectorul $\overrightarrow{CF} = -h \text{ grad } g$. Vectorul grad g este perpendicular la tangenta dusă la curba $g(x_1, x_2) = 0$ în punctul considerat şi orientat în afara domeniului $\mathbf{D}_N$. Vectorul — grad g este opus ca sens lui grad g, deci este orientat spre interiorul domeniului $\mathbf{D}_N$. Înmulţind vectorul —grad g cu constanta h, aleasă astfel încît vîrful vectorului CD să cadă în interiorul domeniului $\mathbf{D}_N$, se obţine efectul dorit.

În punctul D, restricţia $g(x_1, x_2) \leqslant 0$ este satisfăcută, fapt pentru care se execută pasul DG determinat cu relaţia (7—72). Deoarece în punctul G

Scanned with OKEN Scanner

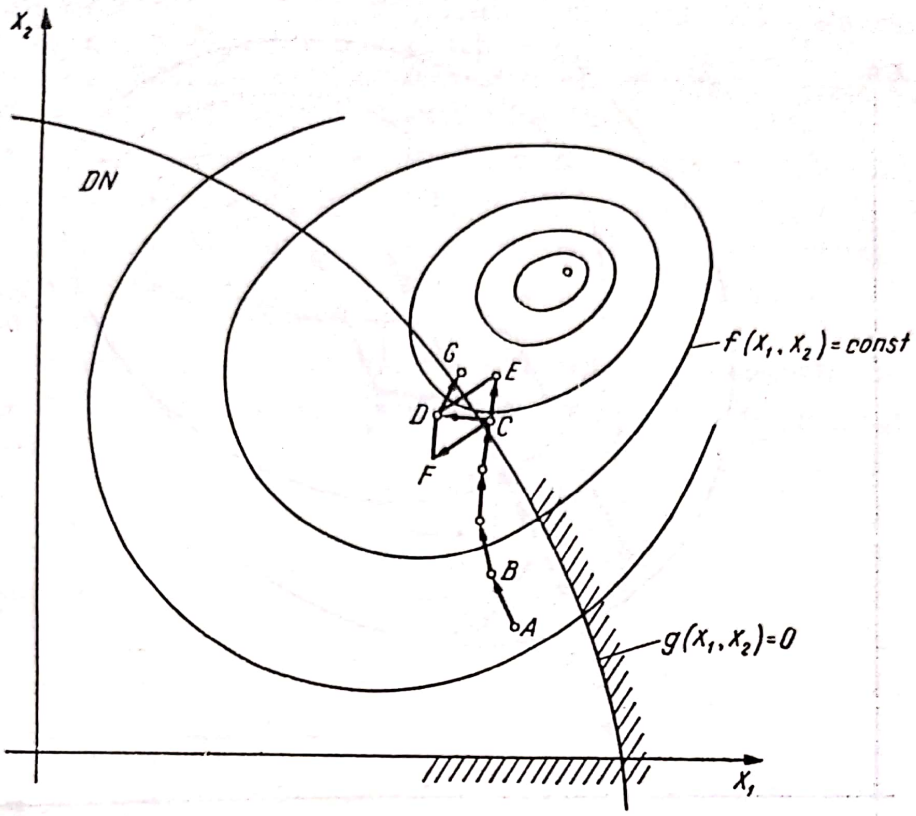


Fig. 7.9

restricția $g(x_1, x_2) \leqslant 0$ nu este satisfăcută, se repetă din nou operațiile efectuate pentru pasul CD.

Pentru cazurile în care restricția $g(x_1, x_2) \leqslant 0$ nu este satisfăcută, relația (7—72) ia forma:

$$\Delta x_i = \pm \frac{\Delta s}{\sqrt{\sum \left(\frac{\partial f}{\partial x_i}\right)^2}} \frac{\partial f}{\partial x_i} - h \frac{\partial g}{\partial x_i} \quad (i = 1, 2). \qquad (7—75)$$

Constanta h nu este adimensională. Din relația (7—75) rezultă că dimensiunea acesteia este determinată de dimensiunea raportului $[(\Delta x_i)^2/\Delta g]$.

Atunci cînd există mai multe restricții neîndeplinite, fie acestea $g_j(x_1, x_2) \leqslant 0$ $(j = 1, 2, \ldots, l_1)$, relația (7—75) devine:

$$\Delta x_i = \pm \frac{\Delta s}{\sqrt{\sum_i \left(\frac{\partial f}{\partial x_i}\right)^2}} \frac{\partial f}{\partial x_i} - h \sum_j \frac{\partial g_j}{\partial x_i}. \qquad (7—76)$$

Ca și în cazul derivatelor parțiale $\partial f/\partial x_i$, aproximate prin diferențele finite (7.73), derivatele parțiale $\partial g_j/\partial x_i$ se aproximează, de asemenea, cu diferențe finite.

Algoritmul simplex. Vom expune principiul acestui algoritm cu ajutorul figurii 7.10, asociată problemei de optim $max f(x_1, x_2)$. În punctele S_{10}, S_{20} și S_{30}, care reprezintă vîrfurile unui triunghi echilateral, se deter-

Scanned with OKEN Scanner

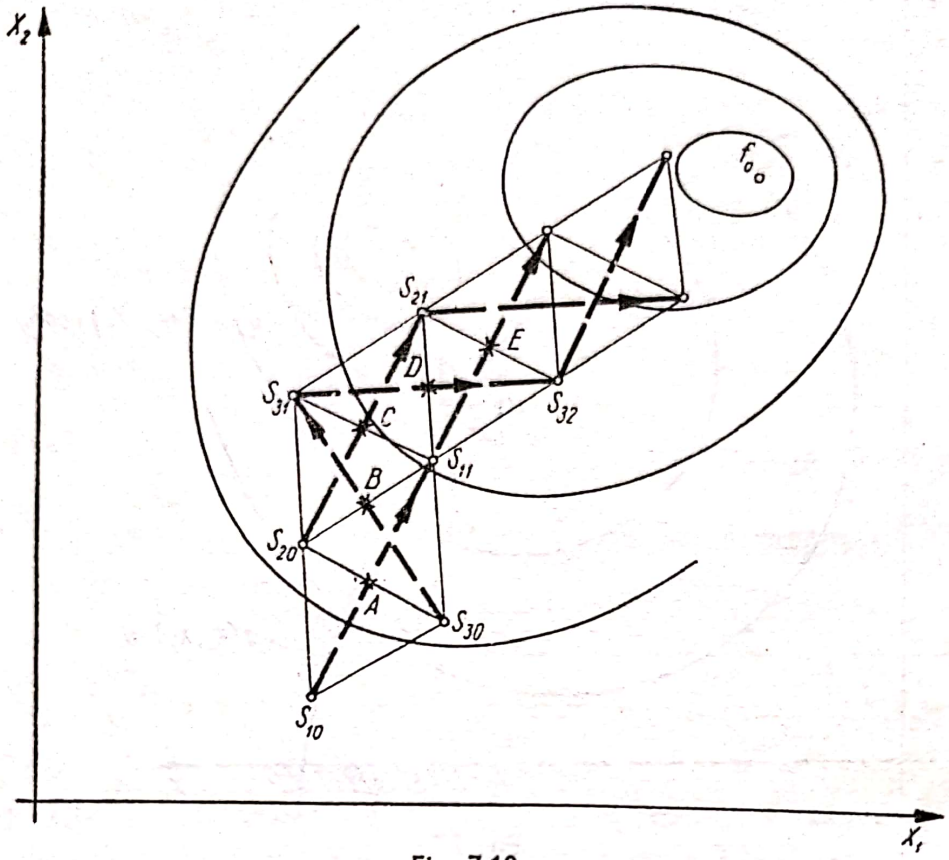


Fig. 7.10

mină valorile funcției de eficiență f_{10}, f_{20} și f_{30}. Fie $f_{10} = min\{f_{10}, f_{20}, f_{30}\}$. Pe latura $S_{20}S_{30}$ se determină punctul A, situat la jumătatea segmentului $S_{20}S_{30}$. Prin punctele S_{10}, A se duce o dreaptă pe care se măsoară segmentul AS_{11}, egal ca mărime cu înălțimea $S_{10}A$ a triunghiului $S_{10}S_{20}S_{30}$. În vîrful S_{11} al triunghiului $S_{20}S_{30}S_{11}$ se determină valoarea funcției de eficiență, f_{11}. Fie, acum, $f_{30} = min\{f_{20}, f_{30}, f_{11}\}$. Pe direcția înălțimii $S_{30}B$ a triunghiului $S_{11}S_{20}S_{30}$ se ia segmentul BS_{31} egal cu $S_{30}B$. În acest mod s-a alcătuit triunghiul $S_{11}S_{20}S_{31}$, pentru care se determină valoarea funcției de eficiență în punctul S_{31} ș.a.m.d.

Continuînd pașii cu aceeași lungime a laturilor triunghiului se ajunge, de obicei, în jurul punctului de maxim, la fenomenul de *ciclizare*. Depășirea acestui impas se face prin micșorarea lungimii laturilor triunghiului. Optimul se consideră atins atunci cînd, prin micșorarea succesivă a laturilor triunghiului, se atinge valoarea minimă aprioric fixată pentru lungimile laturilor triunghiului.

Se observă că pentru problemele de optim cu două variabile independente, figura geometrică obținută în planul acestora este un triunghi. Pentru problemele cu trei variabile independente se obține în spațiul acestora o piramidă dar, în general, pentru probleme cu m variabile, rezultă un poliedru cu $m + 1$ vîrfuri.

Algoritmul explorării. Ideea de bază, pentru acest algoritm, constă în divizarea domeniului de definiție $x_m \leqslant x \leqslant x_M$ al funcției de eficiență, după care, în continuare, se calculează valorile funcțiilor $g_j(x)$, $j=1, 2, \ldots, l$, și ale funcției de eficiență $f(x)$ în fiecare nod al rețelei rezultate în urma divizării efectuate.

Scanned with OKEN Scanner

Pentru fiecare nod al rețelei, înainte de a calcula valoarea funcției de eficiență, se verifică dacă restricțiile $g_j(x) \leqslant 0$, $j = 1, 2, \ldots, l$ sînt îndeplinite. În cazul în care acestea sînt îndeplinite, se trece la calculul funcției de eficiență și apoi la nodul următor.

Dintre valorile funcției de eficiență calculate se reține întotdeauna aceea care este mai mare. În acest mod, la sfîrșitul explorării întregului domeniu $x_m \leqslant x \leqslant x_M$, se obține soluția problemei de optim.

În figura 7.11 este prezentată, în principiu, schema de calcul pentru acest algoritm.

Deși volumul de calcul este mare, pentru problemele de optim cu un număr de variabile relativ mic, acest algoritm este utilizat frecvent în practică.

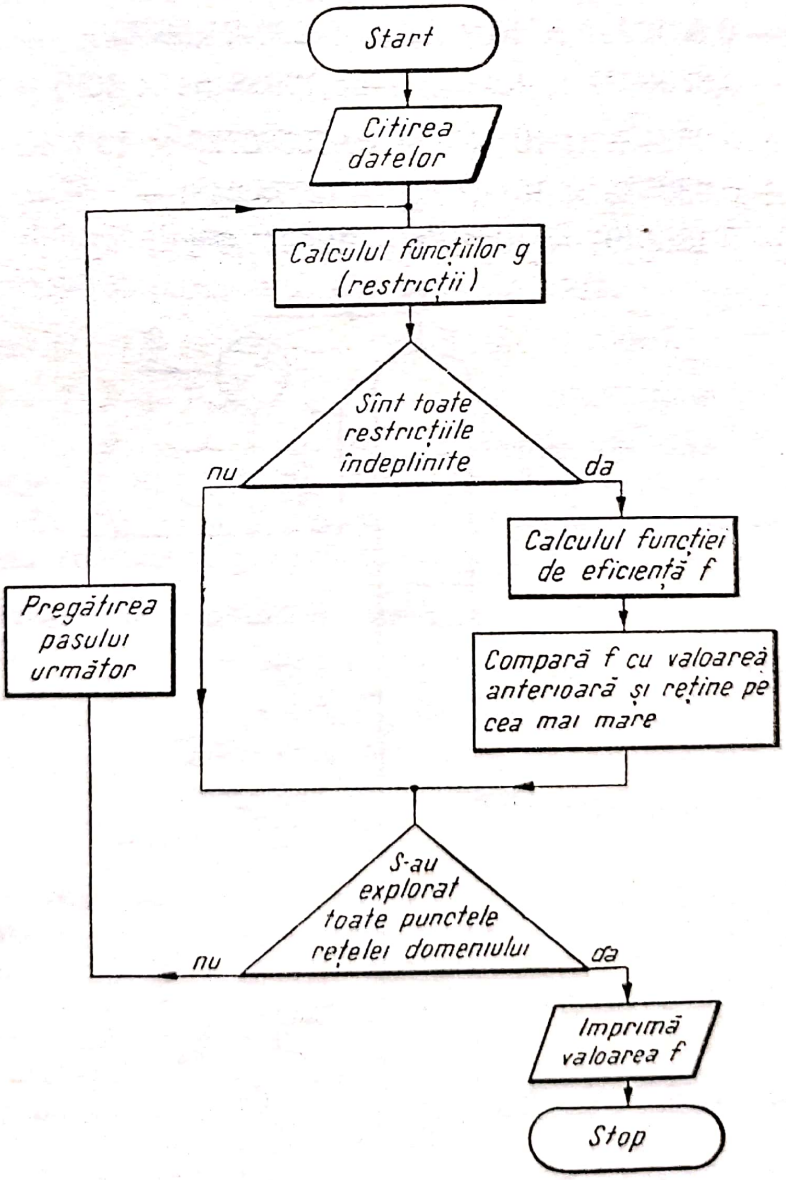


Fig. 7.11

Scanned with OKEN Scanner

Ex. 7—4. Pentru coloana de separare a amestecului etilenă-etan din figura 7.12 s-a obținut următorul model matematic:

$$\begin{aligned} x_p = 0{,}9961 &- 5{,}937243(x_{0E} - 0{,}7941) - 0{,}2575368(T_{0F} + 18{,}0392) - \\ &- 1{,}542722(\pi - 8{,}0059) + 0{,}3377415 \times 10^{-6}(Q_B^* - 3735392{,}1569) - \\ &- 0{,}1601413(x_B - 0{,}0437) + 0{,}7217971\,(q - 0{,}7022) + \\ &+ 0{,}133898 \times 10^{-7}(F^2 - 260200980{,}3922) - \\ &- 0{,}618493 \times 10^{-5}(FT_{0F} + 290421{,}5686) + \\ &+ 0{,}09685217(\pi^2 - 64{,}1084) - \\ &- 0{,}1210659 \times 10^{-12}(Q_B^2 - 0{,}139531545591 \times 10^{14}) + \\ &+ 0{,}1564406 \times 10^{-8}(R^2 - 0{,}09795637 \times 10^{10}) + \\ &+ 0{,}4091646 \times 10^{-3}(Fx_{0F} - 0{,}12775348 \times 10^5); \end{aligned} \tag{7—77}$$

$$\begin{aligned} x_B^* = 0{,}0437 &- 0{,}5812397 \times 10^{-6}(FT_{0F} + 290421{,}5686) - \\ &- 0{,}3606937 \times 10^{-13}(Q_B^2 - 0{,}13953154 \times 10^{14}) + \\ &+ 0{,}258462 \times 10^{-9}(R^2 - 0{,}979563725 \times 10^9) + \\ &+ 0{,}338585 \times 10^{-4}(Fx_{0E} - 12775{,}348) - \\ &- 0{,}1866497 \times 10^{-5}(F\pi - 128941{,}66); \end{aligned}$$

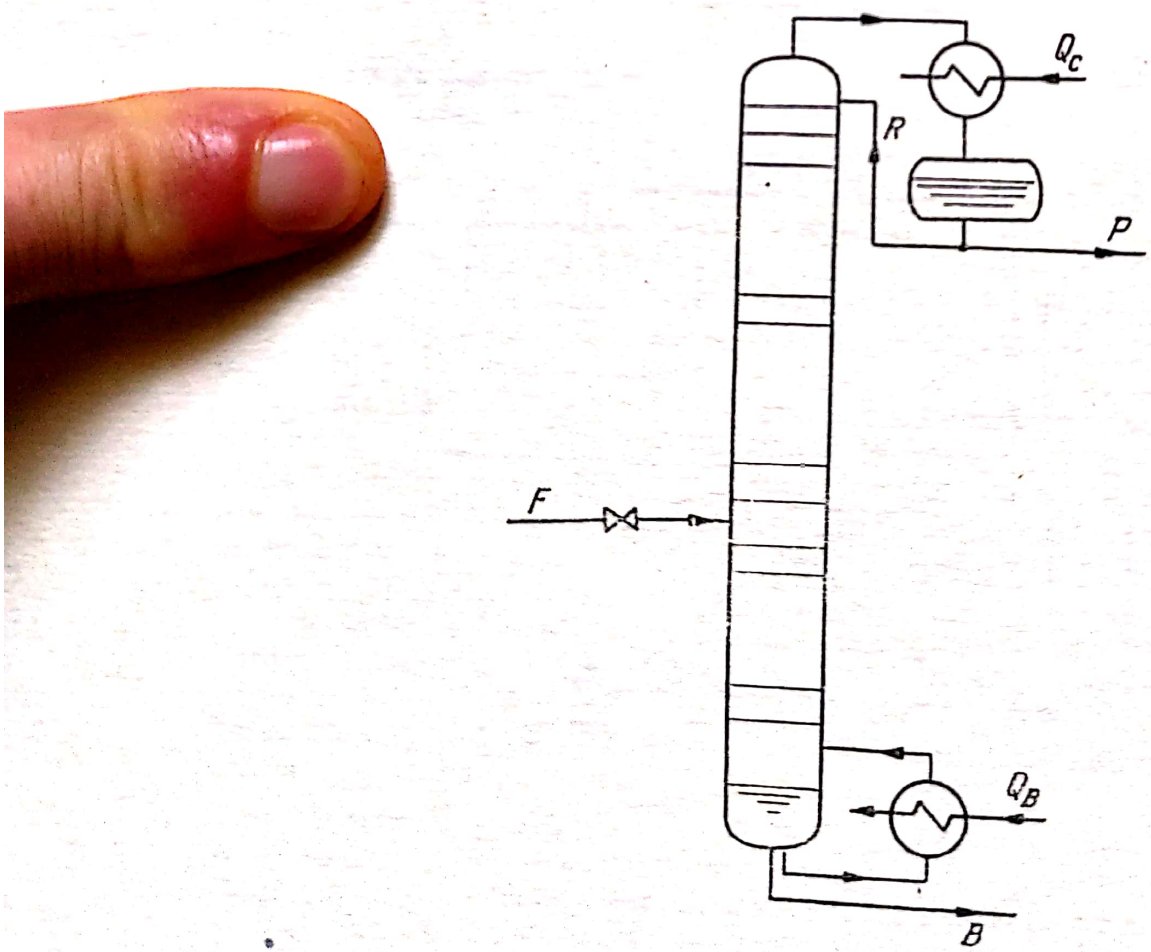


Fig. 7.12

Scanned with OKEN Scanner

$$P = 12501{,}3137 + 0{,}1950364(F - 16107{,}84) +$$
$$+ 0{,}7201976 \times 10^{-2}(FT_{0F} + 290421{,}56) +$$
$$+ 0{,}1178792 \times 10^{-8}(Q_B^2 - 0{,}139531545591 \times 10^{14}) -$$
$$- 0{,}1328589 \times 10^{-4}(R^2 - 0{,}979563725 \times 10^9(Fx_{0F} - 12775{,}348);$$

$$B = F - P,$$

unde:

x_{0F} este concentraţia etilenei în alimentare, în fracţii molare;
T_{0F} — temperatura alimentării înainte de destindere, în °C;
π — presiunea la vîrful coloanei, în daN/cm²;
Q_B — cantitatea de căldură dată la baza coloanei, în kcal/h;
x_B — concentraţia etilenei în baza coloanei, în fracţii molare;
x_p — concentraţia etilenei în produsul de vîrf, în fracţii molare;
q — condiţia termică a alimentării;
F — debitul alimentării, în kg/h;
R — refluxul, în kg/h;
P — debitul produsului de vîrf, în kg/h;
B — debitul produsului din bază, în kg/h.

La modelul (7—77) se asociază funcţia de eficienţă:

$$f = P_1P + P_2B - P_3Q_B - P_4R - P_5F \text{ [lei/h]}, \qquad (7-78)$$

unde $P_1, P_2, \ldots, P_5$ sînt preţurile: produsului de vîrf P, produsului din bază B, căldurii din bază Q_B, refluxului R şi respectiv alimentării F. După cum se observă, această funcţie de eficienţă reprezintă beneficiul* obţinut în urma conducerii procesului de separare.

Se cere rezolvarea următoarei probleme:

$$\begin{gathered} max\, f(R, Q_B); \\ x_B(R, Q_B) > 0; \\ 0{,}999 \leqslant x_p(R, Q_B) < 1; \\ 28060 \leqslant R \leqslant 33000; \\ 0{,}295 \times 10^7 \leqslant Q_B \leqslant 0{,}42 \times 10^7. \end{gathered} \qquad (7-79)$$

Celelalte mărimi care intervin în model, x_{0F}, T_{0F}, π, q şi F, sînt perturbaţii. Pentru ele se presupun anumite valori, astfel că în model rămîn ca variabile independente numai R şi Q_B.

În lista 7.1 este prezentat programul (redactat în FORTRAN), iar în lista 7.2 rezultatul problemei.

Valoarea maximă a funcţiei de eficienţă este $f_{max} = 17\,831$ lei/h şi este obţinută pentru $R = 31\,840$ kg/h şi $Q_B = 0{,}3820 \times 10^7$ kcal/h.

* În mod normal, ar trebui inclusă în funcţia de eficienţă (7—78) şi o constantă care să reprezinte amortizarea, retribuţiile etc. Această constantă nu modifică însă cu nimic soluţia problemei.

Scanned with OKEN Scanner

```
      * DEFINE FILE *1=1,*2=3
      C
      C
      C     LISTA 7.1
      C
      C     PROGRAM PENTRU REZOLVAREA PROBLEMEI DIN EX. 7-4.
      C     CALCULUL SIMPLIFICAT AL OPTIMULUI
      C     COLOANA DE SEPARARE C2H4/C2H6
      C
            DOUBLE PRECISION XOF,TOF,PI,Q,F,XB,XP,P,R,QB,B,FE
            READ (1,10)XOF,TOF,PI,Q,F,M,N
         10 FORMAT(F6.4,F8.4,2F6.4,F10.4,2I5)
            FE1=0.0
            DO 100 I=1,M
            R=28060.0+I*60
            DO 100 J=1,N
            QB=0.295E 07+J*30000
            XB=-0.5812397E-06*F*TOF-0.3606957E-13*QB*QB+0.2584620E-09*R*R
           1+0.5585852E-04*F*XOF-0.1866497E-05*F*PI-0.00057254
            IF (XB.LT.0.0) GO TO 100
            XP=0.9961-5.957243*(XOF-0.7941)-0.2575368*(TOF+18.0592)-1.542722
           1*(PI-8.0059)+0.3377415E-06*(QB-3735392.1569)-0.1601413*(XB-0.0437)
           2+0.7617971*(Q-0.7022)+0.1338987E-07*(F*F-260200980.5922)-0.618493
           3E-05*(F*TOF+290421.5686)+0.0968S217*(PI*PI-64.1084)-0.1210659E-12
           4*(QB*QB-0.139531545591D+14)+0.1564406E-08*(R*R-0.09795637254902
           5D+10)+0.4091646E-03*(F*XOF-0.12775348OD+05)
            IF (XP.GE.1.0.OR.XP.LT.0.999) GO TO 100
            P=0.1950364*F+0.7201976E-02*F*TOF+0.1178792E-08*QB*QB
           1-0.1528589E-04*R*R+0.3754942*F*XOF+3204.734
            B=F-P
            FE=1.725*P+0.7744*B-0.475E-04*QB-0.55E-02*R-0.4*F
            IF(FE1.LT.FE) FE1=FE
            WRITE(3,30)XP,XB,P,FE,B,R,QB
         30 FORMAT(5X,'XP=',F6.4,5X,'XB=',F6.4,5X,'P=',F8.2,5X,'FE=',E12.5,
           15X,'B=',F8.2,5X,'R=',F12.5,5X,'QB=',E14.5)
        100 CONTINUE
            WRITE (3,40) FE1
         40 FORMAT(////10X,'BENEFICIUL MAXIM=',E12.5,'  LEI ORA')
            STOP
            END
```

Lista 7.1

```
XP= .9999  XB= .1494  P= 9919,49  FE= .15150E+05  B= 0188,35  R= .28360E+05  QB= .31500E+07
XP= .9992  XB= .1298  P=10411,17  FE= .15614E+05  B= 5090,67  R= .28900E+05  QB= .32500E+07
XP= .9991  XB= .1047  P=11027,22  FE= .16189E+05  B= 5080,62  R= .29620E+05  QB= .34000E+07
XP= .9991  XB= .0945  P=11275,10  FE= .16420E+05  B= 4832,74  R= .29920E+05  QB= .34600E+07
XP= .9995  XB= .0685  P=11897,00  FE= .17000E+05  B= 4210,84  R= .30700E+05  QB= .36100E+07
XP= .9993  XB= .0524  P=12277,91  FE= .17355E+05  B= 3829,93  R= .31180E+05  QB= .37000E+07
XP= .9991  XB= .0306  P=12789,05  FE= .17831E+05  B= 3318,80  R= .31840E+05  QB= .38200E+07

BENEFICIUL MAXIM= .17831E+05  LEI ORA
```

Lista 7.2

Scanned with OKEN Scanner

```
      * DEFINE FILE *1=1,*2=3
      C
      C
      C     LISTA 7.1
      C
      C     PROGRAM PENTRU REZOLVAREA PROBLEMEI DIN EX. 7-4.
      C     CALCULUL SIMPLIFICAT AL OPTIMULUI
      C     COLOANA DE SEPARARE C2H4/C2H6
      C
            DOUBLE PRECISION XOF,TOF,PI,Q,F,XB,XP,P,R,QB,B,FE
            READ (1,10)XOF,TOF,PI,Q,F,M,N
         10 FORMAT(F6.4,F8.4,2F6.4,F10.4,2I3)
            FE1=0.0
            DO 100 I=1,M
            R=28060.0+I*60
            DO 100 J=1,N
            QB=0.295E 07+J*30000
            XB=-0.5812397E-06*F*TOF-0.3606957E-13*QB*QB+0.2584620E-09*R*R
           1+0.5385852E-04*F*XOF-0.1866497E-05*F*PI-0.00057254
            IF (XB.LT.0.0) GO TO 100
            XP=0.9961-5.957243*(XOF-0.7941)-0.2575368*(TOF+18.0592)-1.542722
           1*(PI-8.0059)+0.3377415E-06*(QB-3735392.1569)-0.1601413*(XB-0.0437)
           2+0.7217971*(Q-0.7022)+0.1338987E-07*(F*F-260200980.3922)-0.618493
           3E-05*(F*TOF+290421.5686)+0.0968217*(PI*PI-64.1084)-0.1210659E-12
           4*(QB*QB-0.1395315455910+14)+0.1364406E-08*(R*R-0.09795637254902
           5D+10)+0.4091646E-03*(F*XOF-0.12775348OD+05)
            IF (XP.GE.1.0.OR.XP.LT.0.999) GO TO 100
            P=0.1950364*F+0.7201976E-02*F*TOF+0.1178792E-08*QB*QB
           1-0.1528589E-04*R*R+0.3754942*F*XOF+3204.734
            B=F-P
            FE=1.725*P+0.7744*B-0.475E-04*QB-0.55E-02*R-0.4*F
            IF(FE1.LT.FE) FE1=FE
            WRITE(3,30)XP,XB,P,FE,B,R,QB
         30 FORMAT(5X,'XP=',F6.4,5X,'XB=',F6.4,5X,'P=',F8.2,5X,'FE=',E12.5,
           15X,'B=',F8.2,5X,'R=',F12.5,5X,'QB=',E14.5)
        100 CONTINUE
            WRITE (3,40) FE1
         40 FORMAT(////10X,'BENEFICIUL MAXIM=',E12.5,'  LEI ORA')
            STOP
            END
```

Lista 7.1

```
XP= .9999   XB= .1494   P= 9919,49   FE= .15150E+05   B= 5188,35   R= .28360E+05   QB= .31500E+07
XP= .9992   XB= .1298   P=10411,17   FE= .15614E+05   B= 5096,67   R= .28900E+05   QB= .32500E+07
XP= .9991   XB= .1047   P=11027,22   FE= .16189E+05   B= 5080,62   R= .29620E+05   QB= .34000E+07
XP= .9991   XB= .0945   P=11275,10   FE= .16420E+05   B= 4832,74   R= .29920E+05   QB= .34600E+07
XP= .9995   XB= .0685   P=11897,00   FE= .17000E+05   B= 4210,84   R= .30700E+05   QB= .36100E+07
XP= .9993   XB= .0524   P=12277,91   FE= .17355E+05   B= 3829,93   R= .31180E+05   QB= .37000E+07
XP= .9991   XB= .0306   P=12789,05   FE= .17831E+05   B= 3318,80   R= .31840E+05   QB= .38200E+07

 BENEFICIUL MAXIM= .17831E+05  LEI ORA
```

Lista 7.2

Scanned with OKEN Scanner

7.3. UN EXEMPLU DE APLICAȚIE
OPTIMIZAREA PROCESULUI DE PIROLIZĂ A ETANULUI

Piroliza constituie procesul de bază pentru petrochimie. Produsele principale care se obțin, etilena și propilena, reprezintă materiile prime pentru numeroase procese petrochimice.

În cele mai multe cazuri, materia primă pentru piroliză este benzina. Pe lîngă aceasta, se mai folosesc și alte materii prime, cum este, de exemplu, etanul. Acesta rezultă fie din procese anterioare, fie chiar din piroliza benzinei, unde, prin recirculare, este condus la unul din reactoarele de piroliză dispuse în paralel.

În figura 7.13 este prezentat schematic un reactor de piroliză. Reactorul propriu-zis se găsește amplasat într-un cuptor de la care primește cantitatea de căldură necesară desfășurării reacțiilor de descompunere ce sînt endoterme.

Variabilele caracteristice procesului sînt:

— perturbații: compoziția materiei prime, temperatura T_i și presiunea la intrare P_i;

— comenzi: debitul de hidrocarburi Q_h, debitul de abur Q_a și temperatura la ieșire T_e;

— ieșiri: compoziția amestecului cracat la ieșirea din reactor.

Introducînd notațiile vectoriale se poate scrie:

$$\mathbf{x}_p = \begin{bmatrix} C_{c2i} \\ C_{c2'i} \\ C_{c3'i} \\ T_i \\ P_i \end{bmatrix}; \quad \mathbf{x}_i = \begin{bmatrix} Q_h \\ Q_s \\ T_e \end{bmatrix}; \quad \mathbf{x}_e = \begin{bmatrix} C_{c2e} \\ C_{c2'e} \\ C_{H2e} \\ C_{c1e} \\ C_{c2''e} \\ C_{c3'e} \\ C_{c4e} \\ C_{c0} \\ P_e \end{bmatrix}. \tag{7—80}$$

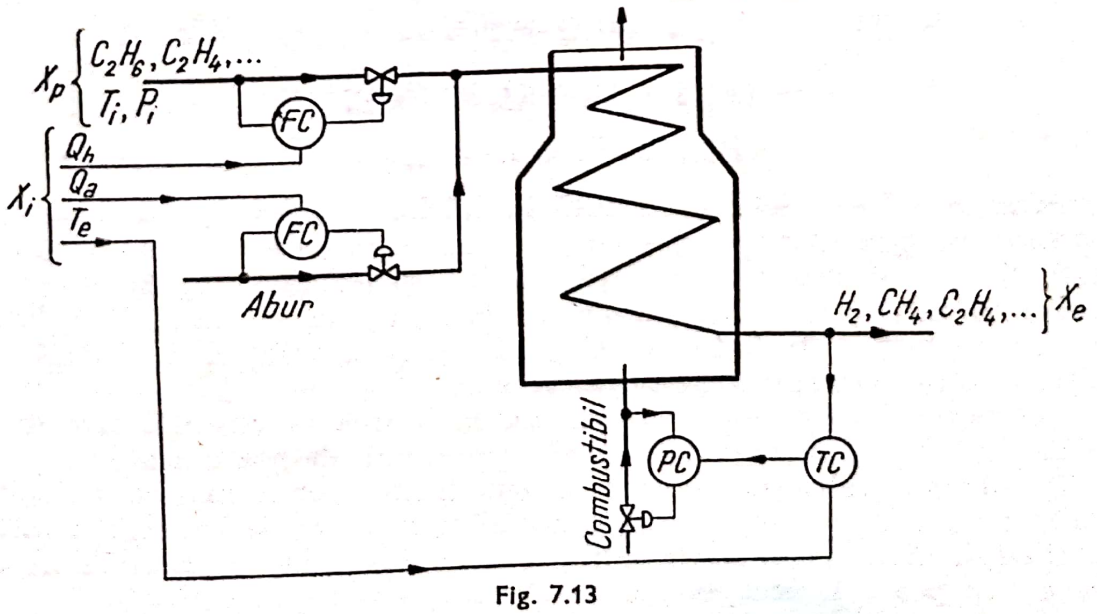


Fig. 7.13

Produsul principal este etilena. Pentru conducerea procesului prezintă o importanţă deosebită obţinerea producţiei de etilenă necesare în condiţii optime. Aceasta implică precizarea funcţiei de eficienţă. Fie beneficiul această funcţie, adică :

$$f = P_1Q_{c2e} + P_2Q_{c2'e} + P_3Q_{H2e} + P_4Q_{c1e} - P_5Q_h - P_6Q_a - P_7Q_T \text{ [lei/h]} \quad (7-81)$$

unde: $P_1, P_2, \cdots, P_7$ sînt preţurile;

$Q_{c2e}, Q_{c2'e}, \ldots, Q_{c1e}$ — debitele de C_2H_6, C_2H_4, H_2 şi CH_4 la ieşirea din reactor, în kg/h ;

Q_T este cantitatea de căldură primită de amestecul reactanţi—produse de reacţie, în kcal/h.

Determinarea maximului funcţiei de eficienţă (7—81) trebuie efectuată în condiţiile prezenţei restricţiilor impuse asupra comenzilor şi mărimilor de ieşire. Matematic, aceasta se exprimă în felul următor :

$$\max_{\substack{Q_h \\ Q_a \\ T_e}} f(P_1Q_{c2e} + P_2Q_{c2'e} + P_3Q_{H2e} + P_4Q_{c1e} - P_5Q_h - P_6Q_a - P_7Q_T) ;$$

$$\begin{aligned}
&P_{11} \leqslant C_{c2e} \leqslant P_{21} ; \\
&P_{12} \leqslant C_{c2'e} \leqslant P_{22} ; \\
&P_{13} \leqslant C_{H2e} \leqslant P_{23} ; \\
&P_{14} \leqslant C_{c1e} \leqslant P_{24} ; \\
&P_{15} \leqslant C_{c2''e} \leqslant P_{25} ; \\
&P_{16} \leqslant C_{c3'e} \leqslant P_{26} ; \\
&P_{17} \leqslant C_{c4e} \leqslant P_{27} ; \\
&P_{18} \leqslant C_{c0e} \leqslant P_{28} ; \\
&P_{19} \leqslant P_e \leqslant P_{29} ; \\
&P_{1,10} \leqslant Q_h + Q_a \leqslant P_{2,10} ; \\
&P_{1,11} \leqslant T_e \leqslant P_{2,11} .
\end{aligned} \quad (7-82)$$

Primele opt restricţii reprezintă limitările de minim şi maxim impuse asupra concentraţiilor produselor şi reactanţilor la ieşirea reactorului. Aceste limite rezultă din condiţiile fizice şi chimice impuse procesului. Exemple în acest sens : o descompunere termică prea adîncă conduce la creşterea concentraţiei de hidrogen şi scăderea concentraţiei de etilenă ; sînt lipsite de sens concentraţiile negative etc.

Restricţia impusă asupra presiunii la ieşirea reactorului este dictată de funcţionarea în bune condiţii a compresorului de gaz cracat.

Penultima restricţie, prezentată sub forma sumei masice a debitelor de materie primă şi abur, rezultă din limitarea vitezei la ieşirea reactorului. Valori ale debitelor peste anumite limite conduc la atingerea vitezei sunetului, ceea ce nu poate fi acceptat în cazul acestui reactor.

Scanned with OKEN Scanner

În fine, ultima restricție asupra temperaturii la ieșirea reactorului este justificată prin aceea că valorile superioare sînt limitate de rezistența tubului, iar valorile inferioare nu sînt interesate pentru reacția chimică principală.

Simularea procesului de piroliză a etanului efectuată în lucrarea [18] a permis exprimarea modelului matematic al reactorului sub forma unor polinoame, în funcție de perturbații și comenzi. Prezentăm aici două exemple (pentru concentrațiile de etan nereacționat și de etilenă):

$$C_{c2e} = 0{,}2590 + 0{,}409458(x_{c3'i} - 0{,}0648) + 0{,}016684(P_i - 3{,}91) - \\ - 0{,}0623169(T_e - 1136{,}4) - 0{,}0969609(Q_a/Q_h - 0{,}3151) + \\ + 0{,}26162 \times 10^{-4}(T_e^2 - 1292056) + 0{,}2769553 \times 10^{-3}(x_{c2i}Q_h - \\ - 1572{,}69) - 0{,}21728 \times 10^{-6}(T_eQ_h - 2041172{,}5); \qquad (7-83)$$

$$C_{c2'e} = 0{,}4830 - 0{,}15289 \times 10^{-3}(T_i - 820{,}67) + 0{,}62401(x_{c2i} - \\ - 0{,}8756) + 1{,}264829(x_{c2'i} - 0{,}0598) - 0{,}059538(P_i - 3{,}91) - \\ - 0{,}273066 \times 10^{-2}(Q_h - 1798) - 0{,}1113589 \times 10^{-2}(Q_a - 556{,}38) + \\ + 0{,}058488(T_e - 1136{,}4) - 0{,}269435 \times 10^{-4}(T_e^2 - 1292056) + \\ + 0{,}2469247 \times 10^{-5}(T_eQ_h - 2041172{,}5) - \\ - 0{,}886627 \times 10^{-6}(T_eQ_a - 632314{,}6). \qquad (7-84)$$

În relațiile (7—83) și (7—84) se observă că fiecare componentă x_{ek} a vectorului de ieșire poate fi calculată cu ajutorul unei expresii de forma:

$$x_{ek} = x_{e0,k} + A_{1k}(T_i - \bar{T}_i) + A_{2k}(P_i - \bar{P}_i) + A_{3k}(x_{c2i} - \bar{X}_{c2i}) + \\ + A_{4k}(x_{c2'i} - \bar{x}_{c2'i}) + A_{5k}(x_{c3'i} - \bar{x}_{c3'i}) + A_{6k}(Q_h - \bar{Q}_h) + \\ + A_{7k}(Q_a - \bar{Q}_a) + A_{8k}(T_e - \bar{T}_e) + A_{9k}(Q_a/Q_h - \overline{Q_a/Q_h}) + \\ + A_{10,k}(Q_h^2 - \bar{Q}_h^2) + A_{11,k}(\overline{Q_a^2} - \bar{Q}_a^2) + A_{12,k}(T_e^2 - \bar{T}_e^2) + \\ + A_{13,k}(T_eQ_h - \overline{T_eQ_h}) + A_{14,k}(T_eQ_a - \overline{T_eQ_a}) + \\ + A_{15,k}(x_{c2i}Q_h - x_{c2i}Q_h), \qquad (7-85)$$

unde prin mărimile notate cu bară deasupra s-au reprezentat valorile medii ale acestor mărimi, rezultate din procedeul analizei de regresie prin care au fost stabilite aceste ecuații.

Introducînd notațiile:

$$A_{01,k} = x_{e0,k} - A_{1k}\bar{T}_i - A_{2k}\bar{P}_i - A_{3k}\overline{x_{c2i}} - A_{4k}\overline{x_{c2'i}} - A_{5k}\overline{x_{c3'i}} - \\ - A_{6k}\bar{Q}_h - A_{7k}\bar{Q}_a - A_{8k}\bar{T}_e - A_{9k}\overline{Q_a/Q_h} - A_{10,k}\overline{Q_h^2} - \\ A_{11}\overline{Q_a^2} - A_{12,k}\overline{T_e^2} - A_{13,k}\overline{T_h} - A_{14,k}\overline{T_eQ_a} - \\ - A_{15,k}\overline{x_{c2i}Q_h}; \qquad (7-86)$$

Scanned with OKEN Scanner

$$A_{02,k} = A_{1k}T_i + A_{2k}P_i + A_{3k}x_{c2i} + A_{4k}x_{c2'i} + A_{5k}x_{c3'i},$$

relaţia (7—85) poate fi pusă sub forma:

$$x_{ek} = A_{01,k} + A_{02,k} + A_{6k}Q_h + A_{7k}Q_a + A_{8k}T_e + A_{9k}Q_a/Q_h + A_{10,k}Q_h^2 + \\ + A_{11,k}Q_a^2 + A_{12,k}T_e^2 + A_{13,k}T_eQ_h + A_{14,k}T_eQ_a + \\ + A_{15,k}x_{c2i}Q_h. \qquad (7—87)$$

Cantitatea de căldură Q_T din funcţia de eficienţă (7—81) este exprimată în funcţie de componentele vectorului x_q şi x_i, printr-o relaţie de forma (7—85). Debitele de etan, etilenă, hidrogen şi metan pot fi exprimate prin produsele dintre debitul masic de hidrocarburi, în kg/h, şi concentraţiile respective, în fracţii masice. Cu aceasta, relaţia (7—81) devine:

$$F = P_1x_{c2e}Q_h + P_2x_{c2'e}Q_h + P_3x_{H2e}Q_h + P_4x_{c1e}Q_h - P_5Q_h - P_6Q_a - \\ - P_7Q_T(Q_h, Q_a, T_e). \qquad (7—88)$$

Înlocuind fiecare concentraţie din relaţia (7—88) cu expresia (7—85) corespunzătoare, funcţia de eficienţă (7—88) ia succesiv formele:

$$f = Q_h\sum_{k=1}^{4}(A_{01,k} + A_{02,k})P_k + Q_h^2\sum_{k=1}^{4}A_{6k}P_a + Q_aQ_h\sum_{k=1}^{4}A_{7k}P_k + \\ + Q_hT_e\sum_{k=1}^{4}A_{8k}P_k + Q_a\sum_{k=1}^{4}A_{9k}P_k + Q_h^3\sum_{k=1}^{4}A_{10,k}P_k + \\ + QhQ_a^2\sum_{k=1}^{4}A_{11,k}P_k + Q_hT_e^2\sum_{k=1}^{4}A_{12,k}P_h + Q_h^2T_e\sum_{k=1}^{4}A_{13,k}P_k + \\ + Q_hQ_aT_e\sum_{k=1}^{4}A_{14,k}P_k + Q_h^2x_{c2i}\sum_{k=1}^{4}A_{15,k}P_k - P_5Q_h - P_6Q_a - \\ - P_7(A_{01,10} + A_{02,10} + Q_hA_{6k} + Q_aA_{7k} + T_eA_{8,10} + Q_a/Q_h \cdot A_{9,10} + \\ + Q_h^2A_{10,10} + Q_a^2A_{11,10} + T_e^2A_{12,10} + T_eQ_hA_{13,10} + T_eQ_aA_{14,10} + \\ + x_{c2i}Q_h \cdot A_{15,10}); \qquad (7—89)$$

$$f = B_1 + B_2Q_h + B_3Q_a + B_4T_e + B_5Q_h^2 + B_6Q_a^2 + B_7T_e^2 + B_8Q_aQ_h + \\ B_9Q_hT_e + B_{10}Q_aT_e + B_{11}Q_h^3 + B_{12}Q_hQ_a^2 + B_{13}Q_hT_e^2 + \qquad (7—90) \\ + B_{14}Q_h^2T_e + B_{15}Q_aQ_hT_e + B_{16}Q_a/Q_h,$$

unde prin $B_1, B_2, \ldots, B_{16}$ s-au notat expresiile corespunzătoare trecerii de la relaţia (7—89) la relaţia (7—90).

Relaţiile (7—87) şi (7—90) sînt folosite în formularea problemei de optim (7—82). În continuare se alcătuieşte problema de programare liniară de forma (7—55), asociată problemei (7—82). Urmărirea acestui proces se poate face relativ uşor, cu ajutorul programului (scris în FORTRAN) prezentat în lista 7.3.

Scanned with OKEN Scanner

```
* DEFINE FILE *1=5,*2=6
C
C LISTA 7.3
C
C         GL     LIMITA DE LINIARIZARE PENTRU X
C         M      NUMARUL DE VARIABILE INDEPENDENTE
C         XO     PUNCTUL IN CARE SE FACE LINIARIZAREA
C         MC     NUMARUL COLOANELOR MATRICEI C
C         NYE    NUMARUL RESTRICTIILOR LIMITATE LA AMBELE CAPETE + FUNCTIA
C                DE EFICIENTA
C         NAC    NUMARUL RESTRICTIILOR Y LIMITATE LA AMBELE CAPETE
C         N      NUMARUL RESTRICTIILOR IMPLICITE SI EXPLICITE
C         KU     VECTOR ASOCIAT RESTRICTIILOR TRANSFORMATE IN EGALITATI
C                   KU(I)=0  RESTRICTIE TIP EGALITATE
C                   KU(I).GT.0  RESTRICTIE TIP INEGALITATE
C         KIY    VECTOR SIMILAR CU KTR; ACESTA RAMINE INSA NEMODIFICAT,
C                CONFORM TIPULUI INITIAL AL RESTRICTIEI
C         KIR    VECTOR CARE INDICA TIPUL RESTRICTIEI
C                 KTR(I)=1 RESTRICTIE TIP Y(I).GE.B(I)
C                 KTR(I)=0 RESTRICTIE TIP Y(I).LE.B(I)
C         NY     NUMARUL RESTRICTIILOR IMPLICITE SI EXPLICITE CU LIMITARE
C                INTR-O SINGURA PARTE
C         NX     NUMARUL COMPONENTELOR VECTORULUI X
C         NP     NUMARUL COMPONENTELOR-PERTURBATII ALE VECTORULUI X
C         K      VECTOR ASOCIAT RESTRICTIILOR Y
C                 K(I)=0 RESTRICTIA Y(I) ESTE SATISFACUTA
C                 K(I)=1 PENTRU CAZURILE IN CARE RESTRICTIA Y(I) ESTE
C                        INCLUSA IN PL SAU, NEFIIND SATISFACUTA, URMEAZA
C                        SA FIE INCLUSA IN PL
C         NE     NUMARUL RESTRICTIILOR DE TIP EGALITATE
C         NKYN   NUMARUL RESTRICTIILOR Y NESATISFACUTE
C         ME     INDICA RESTRICTIILE YL(I).GT.C(I,M1) NEINDEPLINITE SI,
C                C(I,M1) FIIND STRICT POZITIV, SE TRANSFORMA IN EGALITATI
C                PRIN ADAUGAREA UNEI VARIABILE ECART,C(I,M+ME)=-1
C         NATER  INDICA NUMARUL ITERATIILOR
C         NKINT  MARCHEAZA INTOARCERILE DATORITA RESTRICTIILOR CARE NU AU
C                FOST INDEPLINITE
C                  NRINT=0 NORMAL, INAINTE
C                  NRINT=1 INTOARCERE
C         KER    VECTOR PENTRU MARCAREA NEGATIVITATII TERMENULUI LIBER
C                  KER(I)=0 TERMEN LIBER POZITIV
C                  KER(I)=1  TERMEN LIBER NEGATIV
C         P1,P2  LIMITELE INFERIOARA SI SUPERIOARA ALE RESTRICTIILOR Y
C
      COMMON A(30,30),KU(30),MME,MC,L,L1,NE,IND
      DIMENSION K(30),KTR(30),KTY(30),KER(30),
     1          C(30,30),X(30),XO(30),YO(30),Y(30),B(30),GL(30),GLM(30),
     2          AUX(30),AUX(30),XM(30),P1(15),P2(15),A1(15,15),P(15)

      READ(5,10)M,MC,NY,N,NAC,NX,NP,IND,ML,NACYX,DL
   10 FORMAT(10I2,F3,0)
      NP1=NP+1
      ME=0
      ITPR=0
      WRITE(6,11)
   11 FORMAT(/30X,12HDATELE INITIALE/)
      WRITE(6,10)M,MC,NY,N,NAC,NX,NP,IND,ML,NACYX,DL
      READ(5,12)(P(I),I=1,7)
   12 FORMAT(7E11.4)
      WRITE(6,12)(P(I),I=1,7)
      READ(5,20) (XO(I),GL(I),GLM(I),I=NP1,NX)
```


Scanned with OKEN Scanner

```
   20 FORMAT(5E12.5)
      DO 102 I=NP1,NX
  102 X(I)=2.0*GL(I)
      WRITE(6,21)(XO(I),GL(I),GLM(I),I=NP1,NX)
   21 FORMAT(/5(2X,E12.5))
      M1=M+1
      N1=N+1
      NYE=NAC+1
      DO 103 I=1,N1
      KU(I)=0
      KTY(I)=0
  103 KTR(I)=0
      READ(5,22)(P1(I),I=1,NACYX)
   22 FORMAT(8F7.4,F4.2,3F6.1)
      WRITE(6,23)(P1(I),I=1,NACYX)
   23 FORMAT(/12E11.4)
      READ(5,22)(P2(I),I=1,NACYX)
      WRITE(6,23)(P2(I),I=1,NACYX)
      READ(5,24)(KTR(I),I=1,NY)
   24 FORMAT(40I2)
      WRITE(6,25)(KTR(I),I=1,NY)
   25 FORMAT(30(2X,I2))
      DO 104 I=1,N1
  104 K(I)=0
      NITER=-10
      NITAD=0
      DO 119 I=1,NY
  119 KTY(I)=KTR(I)
      DO 105 I=1,N1
      DO 105 J=1,M1
  105 C(I,J)=0.0
      NAC1=NAC+1
C
C     CITIREA DATELOR SI CALCULUL TERMENILOR AO1(I)
C
      READ(5,30)(YO(I),I=1,NAC1)
   30 FORMAT(8F7.5,F5.3,F9.1)
      WRITE(6,31)(YO(I),I=1,NAC1)
   31 FORMAT(/1H ,9(1X,F9.6),1X,F9.1)
      READ(5,32)(XM(I),I=1,NX)
   32 FORMAT(F5.1,3F6.4,F5.2,3F6.1)
      WRITE(6,33)(XM(I),I=1,NX)
   33 FORMAT(/1H ,8E12.5)
      READ(5,34)((A1(I,J),J=1,14),I=1,NAC1)
   34 FORMAT(6E13.6/6E13.6/2E13.6)
      WRITE(6,34)((A1(I,J),J=1,14),I=1,NAC1)
      DO 300 I=1,NAC1
  300 AO1(I)=YO(I)-A1(I,1)*XM(1)-A1(I,2)*XM(2)-A1(I,3)*XM(3)-A1(I,4)*
     1          XM(4)-A1(I,5)*XM(5)-A1(I,6)*XM(6)-A1(I,7)*XM(7)-A1(I,8)*
     2          XM(8)-A1(I,9)*XM(9)/XM(0)-A1(I,10)*XM(6)*XM(6)-A1(I,11)*
     3          XM(6)*XM(8)-A1(I,12)*XM(6)*XM(8)
     4          -A1(I,13)*XM(7)*XM(8)-A1(I,14)*XM(0)*XM(2)
C
C     CITIREA DATELOR SI CALCULUL TERMENILOR AO2(I)
C
      READ(5,40)(X(I),I=1,NP)
   40 FORMAT(6E13.5)
      WRITE(6,40)(X(I),I=1,NP)
      DO 320 I=1,NAC1
  320 AO2(I)=A1(I,1)*X(1)+A1(I,2)*X(2)+A1(I,3)*X(3)+A1(I,4)*X(4)+
     1       A1(I,5)*X(5)
```


Scanned with OKEN Scanner

```
      XO(2)=X(2)
  100 NRYN=0
      NRINI=0
  551 WRITE(6,41)NITER
   41 FORMAT(/10X,9HITERATIA,I5/)
C
C     CALCULUL VALORILOR RESTRICTIILOR Y IN PUNCTUL XO
C
      DO 107 I=1,NAUT
  107 YO(I)=AU1(I)+AU2(I)+A1(I,6)*XU(6)+A1(I,7)*XU(7)+A1(I,8)*XO(8)+
     1      A1(I,9)*XU(7)/XU(6)+A1(I,10)*XU(6)*XO(6)+A1(I,11)*XO(8)
     2      *XU(8)+A1(I,12)*XU(6)*XU(8)+A1(I,13)*XU(7)*XO(8)
     3      +A1(I,14)*XO(2)*XO(6)
C
C     CALCULUL FUNCTIEI DE EFICIENTA IN PUNCTUL XO
C
      B1=-(AU1(10)+AU2(10))*P(5)
      B2=(AU1(1)+AU2(1))*P(1)+(AU1(2)+AU2(2))*P(2)+(AU1(3)+AU1(3))*P(3)+
     1   (AU1(4)+AU2(4))*P(4)-A1(10,6)*P(5)-XU(2)*A1(10,14)*P(5)-P(6)
      B3=A1(1,9)*P(1)+A1(2,9)*P(2)+A1(3,9)*P(3)+A1(4,9)*P(4)-A1(10,7)*
     1   P(5)-P(7)
      B4=-A1(10,8)*P(5)
      B5=A1(1,6)*P(1)+A1(2,6)*P(2)+A1(3,6)*P(3)+A1(4,6)*P(4)-A1(10,10)*
     1   P(5)+XU(2)*(A1(1,14)*P(1)+A1(2,14)*P(2)+A1(3,14)*P(3)+
     2   A1(4,14)*P(4))
      B6=-A1(10,11)*P(5)
      B7= A1(1,7)*P(1)+A1(2,7)*P(2)+A1(3,7)*P(3)+A1(4,7)*P(4)
      B8=-A1(10,13)*P(5)
      B9= A1(1,8)*P(1)+A1(2,8)*P(2)+A1(3,8)*P(3)+A1(4,8)*P(4)-A1(10,12)*
     1   P(5)
      B10=A1(1,10)*P(1)+A1(2,10)*P(2)+A1(3,10)*P(3)+A 1(4,10)*P(4)
      B11=A1(1,12)*P(1)+A1(2,12)*P(2)+A1(3,12)*P(3)+A1(4,12)*P(4)
      B12=A1(1,11)*P(1)+A1(2,11)*P(2)+A1(3,11)*P(3)+A1(4,11)*P(4)
      B13=A1(1,13)*P(1)+A1(2,13)*P(2)+A1(3,13)*P(3)+A1(4,13)*P(4)
      B14=-A1(10,9)*P(5)
      WRITE(6,42) B1,B2,B3,B4,B5,B6,B7,B8,B9,B10,B11,B12,B13,B14
   42 FORMAT(//7(1X,E12.5))
      YERA=B1+B2*XO(6)+B3*XU(7)+B4*XU(8)+B5*XU(6)*XU(6)+B6*XU(8)*XU(8)+
     1     B7*XU(6)*XU(7)+B8*XO(7)*XU(8)+B9*XU(6)*XU(8)+B10*XO(6)**3.0+
     2     B11*XU(6)*XO(6)*XU(8)+B12*XU(6)*XU(6)*XU(8)+B13*XO(6)*XU(7)*
     3     XO(8)+B14*XO(7)/XO(6)
      BF1=B2+2*B5*XU(6)+B7*XU(7)+B9*XU(8)+3*B10*XU(6)*XU(6)+2*B11*XO(6)*
     1     XU(8)+B12*XU(8)*XO(8)+B13*XU(7)*XU(8)-B14*XU(7)/XU(6)**2.0
      BF2=B3+B7*XU(6)+B8*XU(8)+B13*XU(6)*XU(8)+B14/XO(6)
      BF3=B4+2*B6*XU(8)+B8*XU(7)+B9*XU(6)+B11*XO(6)*XU(6)+2*B12*XO(6)*
     1     XO(8)+B13*XO(6)*XO(7)
      DYU3=-BF1*GL(6)-BF2*GL(7)-BF3*GL(8)
      DO 110 I=1,NACYX
      B(I+I-1)=P1(I)
  110 B(I+I)=P2(I)
C
C     CALCULUL ELEMENTELOR MATRICEI C
C
      DO 111 I=1,NAC
      C(I+I-1,1)=A1(I,6)-A1(I,9)*XU(7)/XU(6)+2.0+2*A1(I,10)*XU(6)+
     1     A1(I,12)*XO(8)+A1(I,14)*XO(2)
      C(I+I,1)=C(I+I-1,1)
      C(I+I-1,2)=A1(I,7)+A1(I,9)/XU(6)+A1(I,13)*XU(8)
      C(I+I,2)=C(I+I-1,2)
      C(I+I-1,3)=A1(I,8)+2*A1(I,11)*XU(6)+A1(I,12)*XU(6)+A1(I,13)*XO(7)
      C(I+I,3)=C(I+I-1,3)
```


Scanned with OKEN Scanner

```
      BD=-P0(I)+C(I+1,1)*GL(6)+C(I+1,2)*GL(7)+C(I+1,3)*GL(8)
      C(I+1-1,M1)=P1(I)+BD
  111 C(I+1,M1)=P2(I)+BD
      DO 112 I=NAC1,NACYX
      J=I-NAC1+1
      C(I+1-1,JJ)=1
  112 C(I+1,J)=1
      C(19,M1)=P1(10)-X0(6)+GL(6)
      C(20,M1)=P2(10)-X0(6)+GL(6)
      C(21,M1)=P1(11)-X0(7)+GL(7)
      C(22,M1)=P2(11)-X0(7)+GL(7)
      C(23,M1)=P1(12)-X0(8)+GL(8)
      C(24,M1)=P2(12)-X0(8)+GL(8)
      C(25,1)=1.0
      C(25,M1)=2.0*GL(6)
      C(26,2)=1.0
      C(26,M1)=2.0*GL(7)
      C(27,3)=1.0
      C(27,M1)=2.0*GL(8)
      C(N1,1)=BF1
      C(N1,2)=BF2
      C(N1,3)=BF3
      IF(NITER.GT.5) GO TO 114
C
C     SE IMPRIMA KTR,C,K,KU,X0,Y0 SI YERA
C
      WRITE(6,50)
   50 FORMAT(/40X,10HMATRICEA C/)
      WRITE(6,51)(KTR(I),(C(I,J),J=1,M1),K(I),KU(I),I=1,N1)
   51 FORMAT(I2,3E10.2,2F5.0,E10.2,2I2)
  114 WRITE(6,52)(I,X0(I),I=NP1,NX)
   52 FORMAT(/10X,I2,10X,E14.7)
      NITER=NITER+1
      WRITE(6,52)(I,Y0(I),I=1,NAC1)
      WRITE(6,53)YERA
   53 FORMAT(/5X,32HVALOAREA FUNCTIEI DE EFICIENTA =,E14.7)
      Y0(10)=X0(6)
      Y0(11)=X0(7)
      Y0(12)=X0(8)
      DO 108 I=1,NACYX
      Y(I+1-1)=Y0(I)
  108 Y(I+1)=Y0(I)
      NITAD=NITAD+1
      IF(NITAD.GT.10) GO TO 211
C
C     SE VERIFICA DACA RESTRICTIILE Y0 SINT INDEPLINITE
C
      DO 200 I=1,NY
      IF(K(Y(I).NE.1) GO TO 180
      IF(NRYN.NE.0.AND.K(I).EQ.1) GO TO 172
      IF((Y(I).GE.1.05*B(I)).OR.(Y(I).GE.0.94*B(I))) GO TO 17
      K(I)=1
      NRYN=NRYN+1
      ME=ME+1
      GO TO 172
  171 IF(K(I).NE.1) GO TO 200
      IF(ABS((Y(I)-B(I))/B(I)).LT.0.1) GO TO 172
      K(I)=0
      ME=ME-1
      GO TO 200
  172 IF(C(I,M1).LT.0.0) GO TO 175
      C(I,M+ME)=-1.0
```


Scanned with OKEN Scanner

```
      GO TO 200
  179 KER(I)=1
      C(I,M+ME)=0.0
      KTR(I)=0
      ME=ME-1
      GO TO 200
  180 IF(NRYN.NE.0.AND.K(I).EQ.1) GO TO 182
      IF((Y(I).LE.1.05*B(I)).OR.(Y(I).GE.0.94*B(I))) GO TO 181
      K(I)=1
      NRYN=NRYN+1
      GO TO 182
  181 IF(K(I).NE.1) GO TO 200
      IF(ABS((Y(I)-B(I))/B(I)).LT.0.1) GO TO 182
      K(I)=0
      GO TO 200
  182 IF(C(I,M1).GT.0.0) GO TO 200
      ME=ME+1
      C(I,M+ME)=1.0
      KER(I)=1
      KTR(I)=1
  200 CONTINUE
      MME=M+ME
      IF(NRINT.EQ.1) GO TO 191
      IF(NRYN.EQ.0) GO TO 190
      DO 184 I=NP1,NX
  184 XO(I)=AC(I)-X(I)+GL(I)
      NRINT=1
      GO TO 551
  191 DO 193 I=1,NY
      IF(KER(I).EQ.0) GO TO 193
      DO 192 J=1,M1
  192 C(I,J)=-C(I,J)
  193 CONTINUE
C
C     BLOC PENTRU GENERAREA MATRICEI A SI VECTORULUI B PENTRU PL
C
  190 II=0
      ME=0
      DO 205 I=1,NY
      IF(K(I).NE.1) GO TO 205
      II=II+1
      KU(II)=MME+1
      DO 201 J=1,M1
  201 A(II,J)=C(I,J)
      IF(KTR(I).NE.1) GO TO 205
      KU(II)=0
      ME=ME+1
  205 CONTINUE
      L=II+ME
      L1=L+1
      MC1=MME+1
      NY1=NY+1
      DO 106 I=NY1,M1
      I2=II+I-NY
      KU(I2)=I2+MME
      DO 106 J=1,M1
  106 A(I2,J)=C(I,J)
      DO 107 I=1,I2
  107 A(I,MC1)=A(I,M1)
      IF(NITER.GT.5) GO TO 210
      WRITE(6,27)
```


Scanned with OKEN Scanner

```
   27 FORMAT(//10X,17HMATRICEA A DIN PL//)
      DO 115 I=1,L1
  115 WRITE(6,28)KU(I),(A(I,J),J=1,MC1)
   28 FORMAT(I3,8(1A,E12.5))
      WRITE(6,29)
   29 FORMAT(///25X,20HSOLUTIA PROBLEMEI PL///)
  210 CALL PL
      DO 800 J=1,M
      LU=1000
      DO 801 I=1,L
      IF(KU(I).EQ.J) GO TO 802
  801 CONTINUE
      X(J+NP)=0.0
      GO TO 805
  802 K1=I
      LU=0
  805 IF(LU.NE.1000) X(J+NP)=A(K1,MC1)
  800 CONTINUE
      DO 206 I=1,N1
  206 KU(I)=0
      DO 207 J=NP1,NX
  207 XO(J)=XO(J)+X(J)-GL(J)
      YEL =YERA+DYOS+A(L1,MC1)
      YR=B1+B2*XO(6)+B3*XO(7)+B4*XO(8)+B5*XO(6)*XO(6)+B6*XO(8)*XO(8)+
     1   B7*XO(6)*XO(7)+B8*XO(7)*XO(8)+B9*XO(6)*XO(8)+B10*XO(6)**3.0+
     2   B11*XO(6)*XO(6)*XO(8)+B12*XO(6)*XO(8)*XO(8)+B13*XO(6)*XO(7)*
     3   XO(6)+B14*XO(7)/XO(6)
      WRITE(6,61) YEL
   61 FORMAT(/5X,44HVALOAREA FUNCTIEI DE EFICIENTA LINIARIZATA =,E14.7//)
      WRITE(6,55) YR
      WRITE(6,52) (I,XO(I),I=NP1,NX)
      IF(YR.GT.YERA) GO TO 106
      DO 219 I=NP1,NX
      XO(I)=XO(I)-X(I)+GL(I)
      GL(I)=GL(I)/DL
      IF(GL(I).LT.GLM(I)) GO TO 211
  219 CONTINUE
      GO TO 100
  211 STOP
      END

      SUBROUTINE PL
C        ALGORITMUL SIMPLEX MODIFICAT
C
C
C        C     NUMARUL VARIABILELOR
C        L     NUMARUL RESTRICTIILOR
C        E     NUMARUL EGALITATILOR
C        IND   INDICATOR DE MINIM SAU MAXIM
C                IND=0  PROBLEMA PROVINE DINTR-UNA DE MINIM
C                IND=1  PROBLEMA ESTE DE LA INCEPUT DE MAXIM
C
      COMMON A(30,30),U(30),C,C1,L,L1,E,IND
      INTEGER C,E,C1,X(30),U,FAZA,G
      DO 4 I=1,C
    4 X(I)=I
      L2=L+2
      IF(E) 7,8,7
    8 FAZA=2
      GO TO 15
    7 FAZA=1
      DO 12 J=1,C1
```


Scanned with OKEN Scanner

```
12 A(L2,J)=0.0
   DO 14 I=1,L
   IF(U(I).NE.0) GO TO 14
   DO 13 J=1,C1
13 A(L2,J)=A(L2,J)+A(I,J)
14 CONTINUE
15 W1=-1
   DO 17 J=1,C
   IF(X(J)) 18,17,18
18 LM=L2+1-FAZA
   IF(W1-A(LM,J)) 19,17,17
19 W1=A(LM,J)
   K=J
17 CONTINUE
   IF(W1-0.1E-5) 20,21,21
20 IF(FAZA-1) 22,23,22
23 IF(A(L2,C1)-0.1E-5) 24,25,25
24 FAZA=2
   GO TO 15
25 WRITE(6,26)
26 FORMAT(1H ,5X,26HSISTEMUL ESTE INCOMPATIBIL)
   GO TO 46
22 IF(-W1-0.1E-5) 27,28,28
27 WRITE(6,29)
29 FORMAT(1H ,5X,30HSISTEMUL ARE MAI MULTE SOLUTII)
28 GO TO 41
21 G=0
   DO 30 I=1,L
   IF(A(I,K)) 30,30,32
32 G=G+1
   GENER=A(I,C1)/A(I,K)
   IF(G-1) 33,34,33
33 IF(GENER-G1) 34,30,30
34 G1=GENER
   K1=I
30 CONTINUE
   IF(G) 36,35,36
35 WRITE(6,37)
37 FORMAT(1H ,5X,47HVALOAREA FUNCTIEI DE EFICIENTA ESTE NEMARGINITA)
   GO TO 46
36 GENER=A(K1,K)
   A(K1,K)=0.0
   DO 38 J=1,C1
38 A(K1,J)=A(K1,J)/GENER
   DO 47 J=1,C1
   IF(A(K1,J)) 40,47,40
40 DO 39 I=1,LM
   A(I,J)=A(I,J)-A(K1,J)*A(I,K)
   IF(ABS(A(I,J)).LT.0.1E-5) A(I,J)=0.0
39 CONTINUE
47 CONTINUE
   DO 42 I=1,LM
   A(I,K)=-A(I,K)/GENER
   IF(ABS(A(I,K)).LT.0.1E-5) A(I,K)=0.0
42 CONTINUE
   A(K1,K)=1.0/GENER
   G=X(K)
   X(K)=U(K1)
   U(K1)=G
   GO TO 15
```


Scanned with OKEN Scanner

```
41  DO 43 I=1,L
43  WRITE(6,44) B(I),A(I,C1)
44  FORMAT(T30,I3,T40,E14.7)
    IF(IRD.EQ.1) A(L1,C1)=-A(L1,C1)
    WRITE(6,45) A(L1,C1)
45  FORMAT(//T15,'SUMA PONDERATA FUNCTIEI DE EFICIENTA',T40,E14.7)
46  RETURN
    END
```

Lista 7.3

Scanned with OKEN Scanner

Capitolul 8

APLICAȚIILE PROGRAMĂRII DINAMICE

Așa cum am mai arătat, obiectul metodelor de optimizare îl constituie determinarea maximului sau minimului unei funcții de mai multe variabile.

Cititorului îi sînt cunoscute, desigur, metodele clasice ale calculului diferențial și variațional, de rezolvare a acestei probleme pentru funcții de una sau mai multe variabile, în anumite condiții (în general, fără restricții).

Astfel, una din teoremele de bază ale calculului diferențial, teorema lui Weierstrass, arată că dacă o funcție de o singură variabilă, $f(x)$, este definită și continuă în fiecare punct al unui interval închis, finit, $x \in [a, b]$, atunci în acest interval $f(x)$ atinge o valoare maximă absolută și o valoare minimă absolută.

Există trei variante de căutare a extremului:

1) fie în punctele în care $f'(x) = 0$, dacă $f(x)$ este continuă în intervalul $[a, b]$ și derivabilă în intervalul (a, b), cu excepția unui număr finit de puncte;

2) fie în punctele de la capetele intervalului;

3) fie în punctele în care $f(x)$ nu este derivabilă.

Fără îndoială, cititorul cunoaște „tehnica" determinării practice a punctelor $x_0 \in [a, b]$, în care o funcție dată $f(x)$ prezintă extrem și felul fiecărui extrem $f(x_0)$ — maxim sau minim.

Aflarea extremelor funcțiilor de două variabile apare mai laborioasă; în acest caz, teorema lui Weierstrass arată că fiecare funcție continuă într-un domeniu închis D al variabilelor își atinge o valoare maximă și una minimă în interiorul sau pe conturul domeniului. Ne reamintim că, în cazul unei funcții de două variabile, condiția necesară ca funcția $f(x, y)$ să admită un extrem în punctul $P \in D$ este $f_x = f_y = 0$. Condiția suficientă pentru ca funcția să admită un maxim este:

$$f_{xx} \leqslant 0 \quad \text{cu} \quad f_{xx} f_{yy} - f_{xy}^2 > 0,$$

iar pentru ca să admită un minim:

$$f_{xx} > 0 \quad \text{cu} \quad f_{xx} f_{yy} - f_{xy}^2 > 0.$$

Pentru $f_{xx} \cdot f_{yy} - f_{xy}^2 < 0$, funcția nu are extrem, iar pentru $f_{xx} \cdot f_{yy} - f_{xy}^2 = 0$ trebuie găsit un procedeu particular de aflare a extremului.

Scanned with OKEN Scanner

În sfîrșit, cercetînd problemele aflării extremelor funcțiilor de mai multe variabile, constatăm că lucrurile se pot amplifica atît de mult încît calculul clasic să nu ne mai ofere soluții de găsire practică a acestora.

Ceea ce rămîne valabil în toate cazurile, este *condiția de necesitate*: dacă $F = f(x_1, x_2, \ldots, x_n)$ este diferențiabilă într-un domeniu D, condiția necesară de a avea un extrem într-un punct P din interiorul domeniului este:

$$\frac{\partial f}{\partial x_1} = \frac{\partial f}{\partial x_2} = \ldots = \frac{\partial f}{\partial x_n} = 0,$$

adică $dF = 0$ în punctul $P \in D$.

Pentru anumite forme particulare ale funcției $f(x_1, x_2, \ldots, x_n)$, putem aplica una din metodele clasice ale calculului variațional, ca de exemplu metoda multiplicatorului Lagrange, pentru găsirea valorilor ei extreme.

În numeroase situații practice (cazul conducerii staționar-optimale a proceselor în care apar mai multe etape cu decizii secvențiale), s-a dezvoltat în ultimul timp o metodă eficientă de optimizare cu restricții pentru funcțiile de mai multe variabile, care se bazează pe utilizarea resurselor tehnicilor de calcul automat.

Aceasta este *metoda programării dinamice*, ale cărei aplicații inginerești (mai tipice) formează obiectul acestui capitol.

8.1. INTRODUCERE ÎN PROGRAMAREA DINAMICĂ

Programarea dinamică este o metodă de aflare a maximului sau minimului unei funcții de mai multe variabile. Ea are ca scop, așa cum spune R. Bellman — autorul ei, „să obțină răspunsuri numerice la întrebări numerice" cu ajutorul calculatoarelor automate.

Programarea dinamică are la bază un principiu de optim, enunțat de Bellman [8] astfel:

„O strategie optimă are proprietatea că oricare ar fi starea inițială și decizia inițială, deciziile rămase trebuie să constituie o strategie optimă în raport cu starea care rezultă din prima decizie". Acest principiu se aplică unei largi clase de sisteme optimale.

Deci, pentru a aplica acest principiu, vom împărți procesul de decizie în mai multe etape, considerînd că la fiecare etapă comportamentul viitor al sistemului depinde de starea lui prezentă și nu de comportamentul lui din trecut (sau din „preistorie") și vom căuta decizia optimă în raport cu starea prezentă.

În cele ce urmează, vom înțelege prin *sistem* o funcție împreună cu domeniul ei de definiție, iar prin *proces de decizie* — operațiile de extremizare a funcției, decizia optimă fiind valorile de maximizare sau minimizare a funcției.

Așa cum vom arăta, metoda programării dinamice se reduce în esență la o enumerare și o comparare a unei mulțimi finite (dar numeroase!) de valori.

Să analizăm un exemplu tipic de problemă rezolvabilă prin metoda programării dinamice: maximizarea unei funcții de mai multe variabile supuse unei singure restricții.

Scanned with OKEN Scanner

Fie o resursă x pe care o avem la dispoziție și pe care vrem să o împărțim la N activități, în așa fel încît cele N activități să aducă un venit total maxim. Acest fel de probleme se pun desigur acolo unde resursele sînt limitate și se cere o cît mai bună întrebuințare a lor.

Vom alcătui un model matematic pentru această problemă.

Mai întîi stabilim notațiile: vom nota cu x cantitatea de resursă pe care vrem să o împărțim, iar cu x_i cantitățile parțiale ale resursei x pe care urmează să le determinăm, i fiind indicele care ia valori de la 1 la N și care ne arată activitatea căreia îi va fi alocată resurse x_i.

Să punem în evidență restricțiile:

$$x_i \geqslant 0 \quad (i = 1, 2, \ldots, N),$$

$$x_1 + x_2 + \ldots + x_N = x.$$

Vom nota cu $g_i(x_i)$ venitul pe care îl obținem de la o alocație a unei cantități x_i din resursa x pentru activitatea i. Funcțiile $g_i(x_i)$ au, în general, forma redată în graficul din figura 8.1, formă care este o consecință a două fenomene economice: primul, alocațiile mici aduc venituri mici, iar al doilea, alocațiile prea mari produc un efect de saturație care tinde să egalizeze veniturile. Vom construi modelul pentru acest exemplu luînd în considerare următoarele ipoteze de bază [1]:

1° venitul rezultat dintr-o activitate oarecare este independent de alocațiile pentru alte activități;

2° veniturile provenite de la diferite activități pot fi măsurate cu o unitate comună;

3° venitul total poate fi obținut sub forma de sumă a veniturilor individuale.

În condițiile acestor ipoteze, venitul total al proceselor de alocație are următoarea expresie:

$$R(x_1, x_2, \ldots, x_N) = g_1(x_1) + g_2(x_2) + \ldots + g_N(x_N), \qquad (8-1)$$

pe care, pentru un venit total optim, va trebui să o maximizăm.

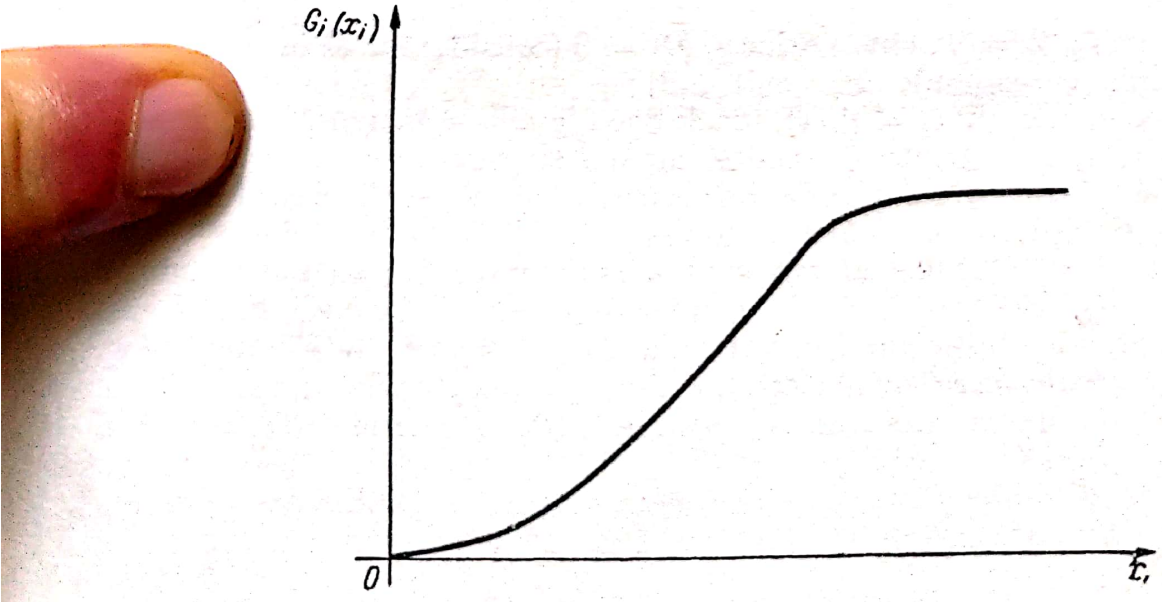


Fig. 8.1

Scanned with OKEN Scanner

Dacă acesteia îi adăugăm restricţiile:

$$\begin{cases} x_i \geqslant 0 \qquad (i = 1, 2, \ldots, N) \\ \sum_{i=1}^{N} x_i = x \end{cases} \tag{8—2}$$

care reprezintă de fapt domeniul funcţiei, obţinem modelul matematic complet al problemei enunţate la început.

Observăm că pentru forme particulare ale funcţiilor g_i se poate folosi cu succes şi calculul clasic — metoda multiplicatorului Lagrange şi metodele programării liniare.

Pentru a maximiza funcţia R prin metoda programării dinamice, adică pentru a putea aplica principiul optimului pe care această metodă îl conţine, vom dota procesul de maximizare cu o proprietate dinamică şi anume: vom pune condiţia ca alocaţiile să fie făcute una cîte una, mai întîi pentru activitatea N, apoi pentru activitatea $N-1$ şi aşa mai departe pînă la activitatea 1.

Procesul de maximizare a funcţiei R, care la prima vedere părea static, a fost transformat astfel într-un proces dinamic.

Vom pune acum în evidenţă dependenţa funcţiei $R(x_1, x_2, \ldots, x_n)$ de restricţia $\sum_{i=1}^{N} x_i = x$ şi de N. Vom introduce şirul de funcţii $f_N(x)$ definit astfel:

$$f_N(x) = \max_{\{x_i\}} R(x_1, x_2, \ldots, x_n). \tag{8—3}$$

Astfel, funcţia $f_N(x)$ este venitul optim de la o alocaţie a cantităţii de resurse x pentru N activităţi, funcţia $f_{N-1}(x)$ este venitul optim de la o alocaţie a cantităţii de resurse x pentru $N-1$ activităţi ş.a.m.d.

Vom vedea cum se poate transforma, cu ajutorul acestor şiruri, o problemă de maximizare a unei funcţii de n variabile într-o problemă de maximizare consecutivă a unei funcţii de o variabilă.

Termenii şirului $\{f_N(x)\}$ iau, în două cazuri, valori care se obţin imediat:

— evident $f_N(0) = 0$ pentru orice $g_i(0) = 0$ (considerăm că la o alocaţie de resursă nulă corespunde un venit nul);

— de asemenea $f_1(x) = g_1(x)$, căci dacă avem o singură activitate, vom aloca întreaga cantitate de resurse acestei singure activităţi.

Pentru a obţine o relaţie de recurenţă între termenii şirului $\{f_N(x)\}$ vom aplica principiul optimizării dinamice astfel:

Fie $x_N \in [0, x]$ alocaţia aleasă pentru activitatea N; ea ne va aduce un venit $g_N(x_N)$. Se ştie însă că restul cantităţii de resursă, $x - x_N$, va fi folosit pentru a obţine un venit maxim de la celelalte $N-1$ activităţi. Acest venit maxim este prin definiţie $f_{N-1}(x - x_N)$ care, împreună cu venitul provenit de la alocarea x_N pentru a N-a activitate, dă un venit total $g_N(x_N) + f_{N-1}(x - x_N)$.

O alegere optimă a lui x_N este alegerea care maximizează această funcţie. Obţinem astfel ecuaţia fundamentală a programării dinamice:

$$f_N(x) = \max_{x_N \in [0,x]} [g_N(x_N) + f_{N-1}(x - x_N)]. \tag{8—4}$$

Scanned with OKEN Scanner

Aceasta este o ecuaţie de recurenţă la care primul termen al şirului are expresia :

$$f_1(x) = g_1(x). \qquad (8-5)$$

Vom intra în detalii pentru a vedea cum putem afla maximul expresiei (8—4) folosind un sistem de calcul automat.

Observăm, de la început, că avem trei mărimi care variază în următoarea ordine :

x_i în intervalul $[0, x]$,
x în intervalul $[0, X]$ şi
i în intervalul $[1, N]$,

unde : X este cantitatea totală maximă a resursei x, fixată, iar N este numărul total de activităţi (de asemenea fixat).

Pentru a putea lucra în cazul unui calculator cu funcţiile g_i şi f_i, vom folosi valorile lor discretizate şi tabelate. Pentru aceasta, este necesar ca argumentele x şi x_i să varieze pe o reţea de puncte $0, \Delta, 2\Delta, \ldots . M\Delta$, Δ fiind pasul reţelei [7]. Mărimea lui Δ se alege în funcţie de precizia cerută calculului şi memoria disponibilă, şi este convenabil să fie aceeaşi şi pentru x şi pentru x_i.

În funcţie de precizia cerută rezultatelor, vom folosi interpolări liniare [7] sau neliniare la evaluarea funcţiilor f_i şi g_i.

Datele de intrare sînt :

N — numărul de activităţi ;
X — valoarea maximă a resursei totale ;
$g_i(x)$ — valorile tabelate ale funcţiilor venit.

Procesul de maximizare începe cu prima activitate. Se evaluează funcţia $f_1(x) = g_1(x)$, adică :

$$f_1(k\Delta) = g_1(k\Delta) \qquad (k = 0, 1, 2, \ldots, M). \qquad (8-6)$$

Urmează apoi activitatea a doua. De data aceasta se evaluează funcţia :

$$f_2(x) = \max_{x_2 \in [0,x]} [g_2(x_2) + f_1(x - x_2)],$$

adică :

$$f_2(x) = \max_{k=0,1,\ldots,M} [g_2(k\Delta) + f_1(x - k\Delta)], \qquad (8-7)$$

unde x va parcurge la rîndul lui întreg şirul de valori discrete date de k, pe care le-a parcurs şi x_2.

Evaluarea funcţiei $f_2(x)$ înseamnă de fapt reţinerea în tabela $f_2(x)$, pentru fiecare valoare a lui x, a valorii maxime a expresiei din membrul drept al ecuaţiei (8—7).

Este de asemenea, reţinută şi valoarea acelui x_2 la care s-a produs maximul într-o tabelă asociată $x_2(x)$.

Practic se porneşte cu $x = 0$ şi se calculează $g_2(0) + f_1(x)$ şi $g_2(\Delta) + f_1(x - \Delta)$. Apoi se compară cele două valori şi se reţine cea mai mare dintre ele, după care se calculează valoarea $g_2(2\Delta) + f_2(x - 2\Delta)$, care este comparată cu valoarea mai mare obţinută anterior şi se reţine cea mai mare dintre ele. Această operaţie continuă pînă ce k a trecut prin toate valorile permise. Altfel spus, x_2 se măreşte la fiecare pas cu Δ pînă ce devine

Scanned with OKEN Scanner

cel mult egal cu x, după care x se mărește cu Δ, iar x_2 reia valoarea zero și continuă să se mărească cu Δ pînă ce atinge noua valoare a lui x.

Procedeul continuă pînă ce x a parcurs toate valorile de la zero la X. Odată încheiată această fază, ne aflăm în posesia tabelei $f_2(x)$ precum și a unei tabele $x_2(x)$ corespunzătoare.

Trecînd la activitatea a treia, vom avea de evaluat funcția:

$$f_3(x) = \max_{x_3 \in [0,x]} [g_3(x_3) + f_2(x - x_3)],$$

adică:

$$f_3(x) = \max_{k=0.1,\ldots,M} [g_3(k\Delta) + f_2(x - k\Delta).$$

În procesul de căutare a maximului, x va porni din nou de la valoarea zero și va ajunge la valoarea X, iar x_3 va evolua între valorile zero și x. Vom obține și de data aceasta o tabelă $f_3(x)$ și o tabelă $x_3(x)$ corespunzătoare ei, care ne va da pentru fiecare valoare a lui x, valorile x_3 ce maximează funcția $f_3(x)$.

Vom continua astfel calculele pînă la activitatea N. La fiecare va trebui să păstrăm în memoria calculatorului (internă sau externă, după posibilitățile sistemului și necesitățile de calcul) tabelele $x_i(x)$, deoarece ele ne vor da „decizia optimă".

Pentru obținerea deciziei optime vom alege o resursă x, de obicei cea maximă X, și vom citi în tabela $x_N(x)$ valoarea lui x_N; vom micșora resursa x cu cantitatea x_N, adică $x := x - x_N(x)$ și cu această cantitate vom citi în tabela $x_{N-1}(x)$ valoarea lui x_{N-1}; vom continua procedeul pînă la aflarea lui x_1. Șirul $x_1, x_2, \ldots, x_N$ reprezintă soluția problemei.

Observăm că în această fază a calculelor, tabelele $f_i(x)$ nu mai sînt necesare, rolul lor încheindu-se după determinarea tabelelor $x_i(x)$. De asemenea, observăm că fiind în posesia tabelelor $x_i(x)$, există mai multe seturi de decizii optime, fiecare set corespunzînd cantității totale de resursa x cu care se pornește de la tabela $x_N(x)$.

În figura 8.2 este reprezentată schema de calcul corespunzătoare operațiilor descrise mai sus. În această schemă au fost folosite simbolurile specifice limbajului FORTRAN pentru operațiile aritmetice și pentru atribuire, adică:

* pentru înmulțire,
/ pentru împărțire,
= pentru := („ia valoarea").

8.2. EXEMPLE DE APLICAȚII

Pentru a da posibilitate cititorului să aplice efectiv programarea dinamică de tipul descris în § 8.1, vom prezenta în continuare trei exemple concrete, cu formularea problemei, algoritmului și programului de calcul aferent (redactat în FORTRAN).

Ex. 8—1. Să considerăm mai întîi problema încărcării unui vapor cu mașini și utilaje petroliere de diferite tipuri, destinate exportului.

Scanned with OKEN Scanner

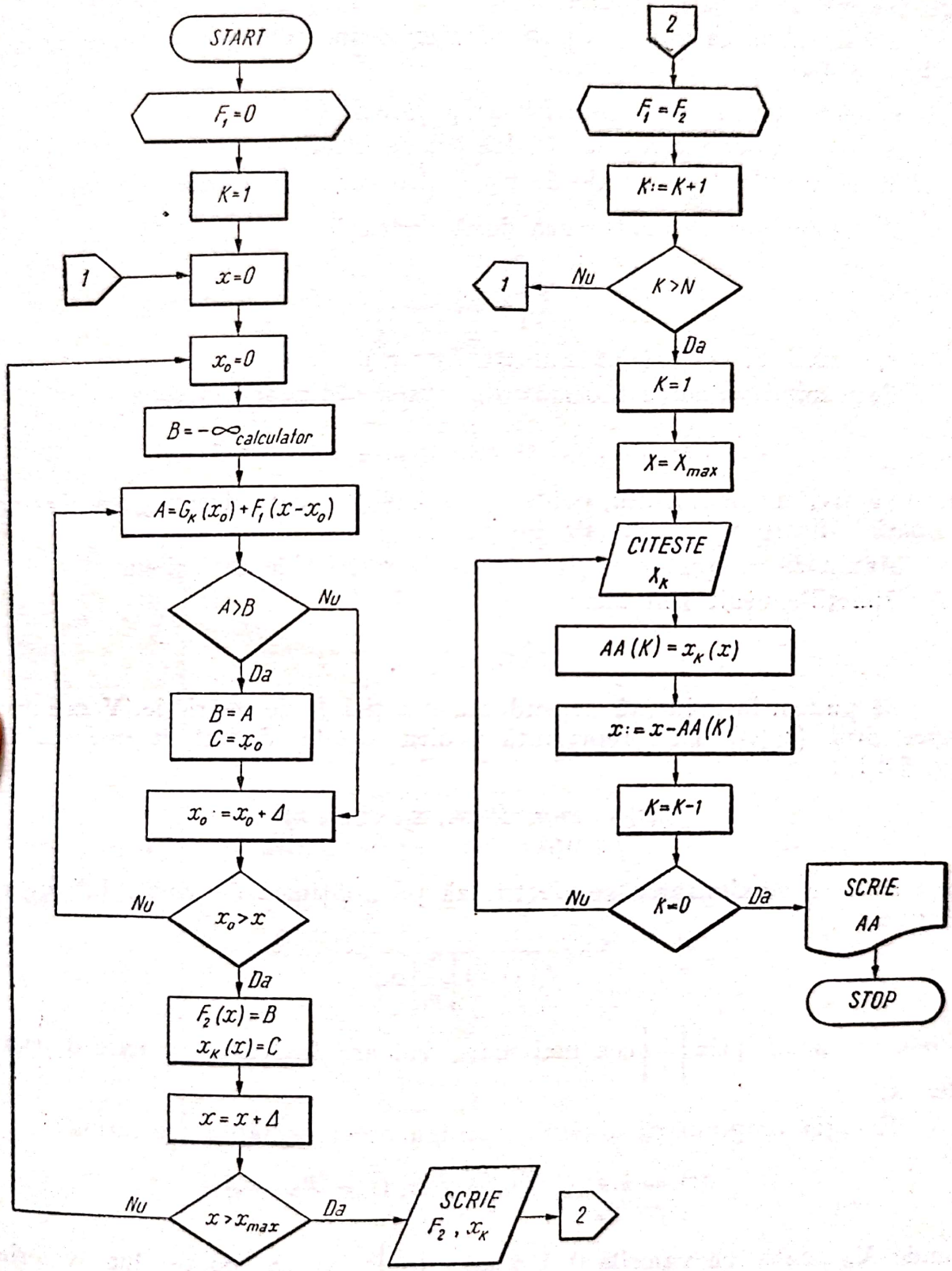


Fig. 8.2

Scanned with OKEN Scanner

Se pune problema cum vom încărca un vas de capacitate limitată cu o încărcătură de valoare maximă? Prin *valoare* putem înțelege uneori urgența de punere în funcțiune etc.

Presupunem că avem i tipuri de utilaj de încărcat, pe un vas de capacitate z, și fie:

v_i — valoarea celui de-al i-lea tip de utilaj;
p_i — greutatea celui de al i-lea tip de utilaj;
x_i — numărul de utilaje de tip i care vor fi încărcate.

De la început se conturează două restricții:

$$\sum_{i=1}^{N} x_i p_i \leqslant z,$$

$x_i \in \{0, 1, 2, \cdots\}$ (sînt numere întregi).

Exprimînd valoarea încărcăturii totale, obținem funcția:

$$R(x_1, x_2, \ldots, x_N) = x_1 v_1 + x_2 v_2 + \ldots + x_N v_N,$$

care va trebui maximizată, evident, în cazul în care dorim „cea mai valoroasă" dintre variantele de încărcare.

Maximizarea trebuie să țină seama de restricțiile de mai sus,

Funcțiile venit sînt aici:

$$g_i(x_i) = x_i v_i.$$

Să punem în evidență dependența funcției R de restricție. Vom introduce șirul $\{f_N(z)\}$, care reprezintă venitul optim definit ca mai înainte (v. § 8.1):

$$f_N(z) = \max_{\{x_i\}} R(x_1, x_2, \ldots, x_N);$$

pentru care maximizarea se efectuează pe mulțimea de valori întregi x_i. Atunci:

$$f_1(x) = \left[\frac{z}{p_1}\right] v_1,$$

unde am notat prin $\left[\frac{z}{p}\right]$ cea mai mare valoare întreagă pe care o poate lua x.

Ecuația programării dinamice pentru acest exemplu are forma:

$$f_N(z) = \max_{x_N} \{x_N v_N + f_{N-1}(z - X_N \cdot p_N)\},$$

unde X_N poate lua valorile $0, 1, 2, \ldots, \left[\frac{z}{p}\right]$, iar z poate lua valorile $0, 1, 2, \ldots, W$ (unde W este capacitatea maximă).

Schema de calcul pentru rezolvarea problemei de programare dinamică din *Ex. 8—1* este prezentată în figura 8.3, iar programul ce rezultă din

Scanned with OKEN Scanner

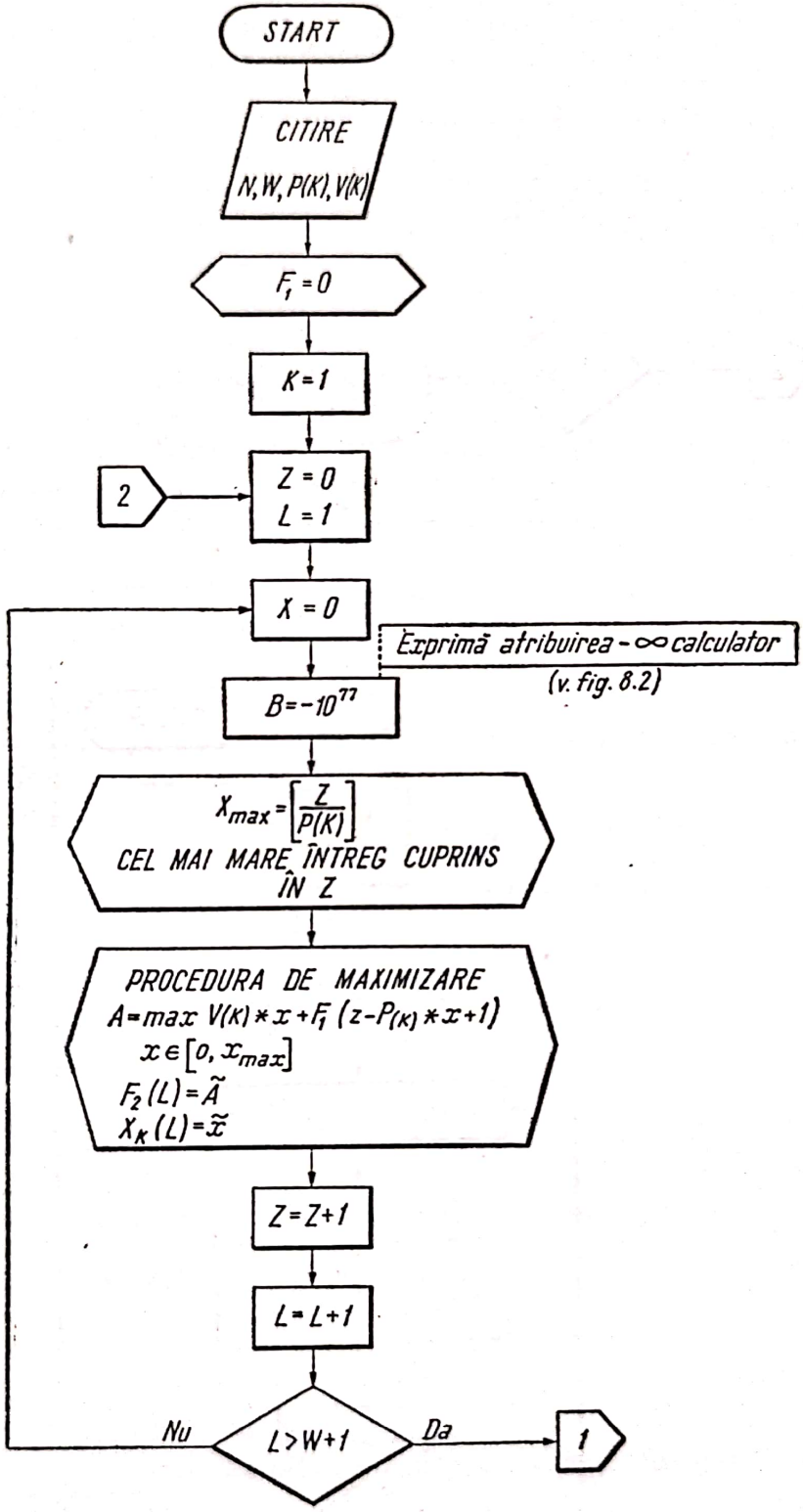
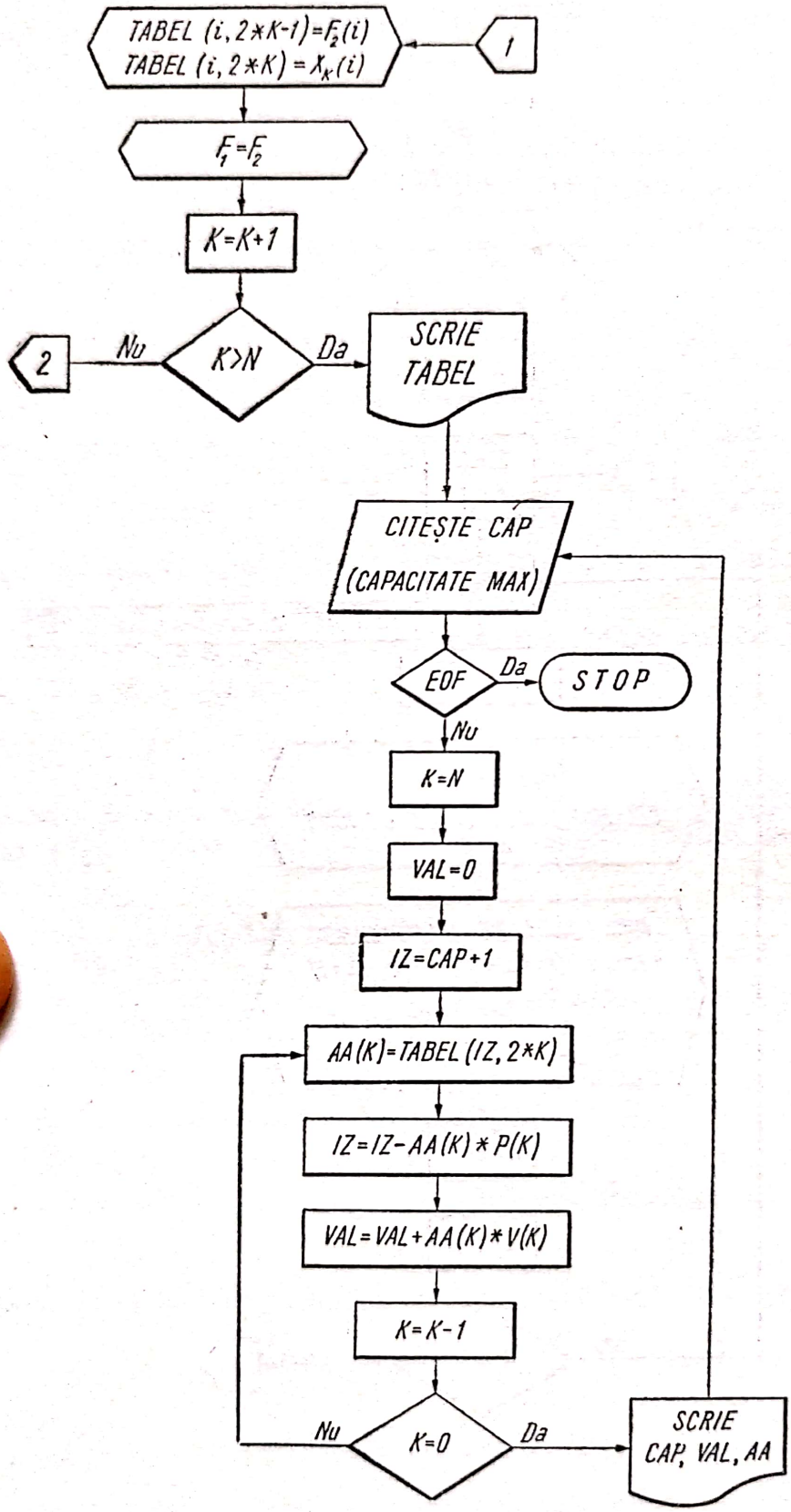


Fig. 8.3

Scanned with OKEN Scanner



Scanned with OKEN Scanner

această schemă, scris în FORTRAN, este redat în lista 8.1. Rezultatele obținute cu acest program, pe baza datelor de intrare din lista 8.2 și pentru tabelele de lucru din lista 8.3, sînt indicate în „out-print'-ul din lista 8.4.

*
* *

În problema precedentă am folosit metoda programării dinamice pentru a repartiza o resursă x la N activități.

În cele ce urmează vom extinde metoda programării dinamice pentru a putea repartiza **două tipuri de resursă,** x și y, la N activități.

În principiu, programarea dinamică se poate aplica pentru a repartiza M tipuri de resursă la N activități, dar există posibilitatea de a depăși rapid capacitatea memoriei calculatorului. Dimensiunile tabelelor g_i și f_i devin $(L+1)^M$, unde L este numărul de puncte pe rețea, pe care l-ar parcurge resursa X, iar M este numărul de resurse diferite.

Programarea dinamică poate fi folosită totuși și în cazurile arătate mai sus, dar aplicînd anumite procedee matematice, care vor evita creșterea dimensiunilor tabelelor (multiplicatorul Lagrange, aproximații succesive etc. [1]).

Să notăm cu x și y cele două feluri de resurse, iar cu x_i și y_i cantitățile respective, de repartizat pentru activitatea i.

Fie funcția $g_i(x_i, y_i)$ venitul provenit de la activitatea i, cînd s-au făcut repartițiile de resurse x_i, y_i.

Cu aceste notații putem trece la rezolvarea unei probleme absolut analoage celei rezolvate mai înainte.

Avem de maximizat funcția:

$$R(x_1, x_2, \ldots, x_N; y_1, y_2, \ldots, y_N) = \sum_{i=1}^{N} g_i(x_i, y_i),$$

supusă restricțiilor:

$$\sum_{i=1}^{N} x_i = x \text{ cu } x_i \geqslant 0,$$

$$\sum_{i=1}^{N} y_i = y \text{ cu } y_i \geqslant 0.$$

Introducem șirul de funcții numite venitul optim:

$$f_N(x, y) = \max_{\{x,y\}} R_N(x_1, x_2, \ldots, x_N; y_1, y_2, \ldots, y_N).$$

Maximizarea se efectuează pe domeniul valorilor x_i și y_i nenegative, cu N variind de la 1, 2, 3 ... înainte.

Pentru $N = 1$, $f_1(x, y) = g_1(x, y)$.

Pentru $N \geqslant 2$ scriem ecuația fundamentală a programării dinamice astfel:

$$f_N(x, y) = \max_{0 \leqslant x_N \leqslant x} \{ \max_{0 \leqslant y_N \leqslant y} [g_N(x_N, y_N) + f_{N-1}(x - x_N, y - y_N)] \}.$$

Să considerăm un caz particular al problemei bidimensionale de mai sus: fie x_i și y_i legați prin relația de forma $y_i = h_i(x_i)$.

Scanned with OKEN Scanner

```
C        LISTA 8.1
C        PROGRAMARE DINAMICA
C        PROGRAM MONODIMENSIONAL PENTRU FUNCTII DE N VARIABILE
C     (EX.8-1: INCARCARE UTILAJE CU RESTRICTII DE GREUTATE)
C
      DIMENSION P(10),V(10),F1(101),F2(101),XK(101),AA(10),TABEL(101,20)
      DATA BB/Z880000000/
C
C        CITIREA PARAMETRILOR :  NUMAR UTILAJE                    N
C                                CAPACITATE MAXIMA                W
C                                GREUTATI SI VALORI UTILAJE       P,V
C        K INDICELE UTILAJULUI
C        L INDICELE CAPACITATII (L=Z+1)
C
      READ (105,200) N,W,(P(I),I=1,N),(V(I),I=1,N)
C     SCRIERE DATE INITIALE
      WRITE (108,400)  N,W,(I,P(I),V(I),I=1,N)
      WRITE(108,401) (I,I,I=1,N)
C
      NZ=W+1
      DO 1 I=1,NZ
    1 F1(I)=0.
      DO 7  K=1,N
      Z=0
      DO  6 L=1,NZ
      X=0
      B=BB
C          PROCEDURA DE AFLARE A CELUI MAI MARE INTREG CUPRINS IN Z
      I=0
      XMAX=Z
    2 IF(XMAX.LT.P(K)) GO TO 3
      XMAX=XMAX-P(K)
      I=I+1
      GO TO 2
C
    3 XMAX=I
C     INDICELE DE INTRARE IN TABELA F1
    4 IND=Z-P(K)*X+1
      A=V(K)*X+F1(IND)
      IF (A.LE.B) GO TO 5
      B=A
      C=X
    5 X=X+1
      IF (X.LE.XMAX) GO TO 4
C
      XK(L)=C
      F2(L)=B
    6 Z=Z+1
      DO 7 I=1,NZ
C        MEMORARE IN TABEL PENTRU IMPRIMARE XK SI F2
      TABEL(I,2*K-1)=F2(I)
      TABEL(I,2*K  )=XK(I)
    7 F1(I)=F2(I)
      N2=2*N
      DO 8 I=1,NZ
      Z=I-1
      WRITE(108,402)  Z,(TABEL(I,J),J=1,N2)
```


Scanned with OKEN Scanner

```
    8 CONTINUE
C
C         EXPLORAREA TABELELOR
C
C         CITIRE CAPACITATE VAS
      WRITE(108,403)(I,I=1,N)
    9 READ(105,200,END=11) CAP
      K=N
C         IZ INDICELE CAPACITATII DISPONIBILE
      IZ=CAP+1
      VAL=0
   10 AA(K)=TABEL(IZ,2*K)
      IZ=IZ-AA(K)*P(K)
C          VALOARE CUMULATA
      VAL=VAL+AA(K)*V(K)
      K=K-1
      IF (K.GT.0) GO TO 10
      WRITE(108,404) CAP,VAL,(AA(I),I=1,N)
      GO TO 9
   11 STOP
C
  200 FORMAT(I5,F5.0/(20F4.0))
  400 FORMAT ('1',10X,'DATELE DE INTRARE:'/10X,5(4H****)/
     *'0   NUMARUL UTILAJELOR    ',I10/'     CAPACITATE (TONE)',I14
     */'0   N.UT.        GREUTATE      VALOARE   '/(I7,2F14.0))
  401 FORMAT ('1',30X,'TABELELE DE LUCRU FK(X) SI XK(X):'/30X,35(1H*)/
     *'0     Z ', 7('    F',I2,'    X',I2))
  402 FORMAT(F6.0,10(F6.1,I6))
  403 FORMAT ('1',16X,'TABELA REZULTATELOR:'/16X,22(1H*)/
     *'0 CAP.TOT. VALOARE',10X,'N U M A R  U T I L A J'/20X,7I5/)
  404 FORMAT (2(F7.0,3X),10I5)
C
      END
```

Lista 8.1

```
                DATELE DE INTRARE:
               ********************

NUMARUL UTILAJELOR                 7
CAPACITATE (TONE)                 30

N.UT.          GREUTATE       VALOARE
  1              3.            100.
  2              9.            450.
  3              7.            435.
  4              9.            480.
  5              8.            435.
  6              7.            425.
  7              1.             20.
```

Lista 8.2

Scanned with OKEN Scanner

TABELELE DE LUCRU FK(X) SI XK(X):

****

Z	F 1	Y 1	F 2	X 2	F 3	X 3	F 4	X 4	F 5	X 5	F 6	X 6	F 7	X 7
0.	.0	0	.0	0	.0	0	.0	0	.0	0	.0	0	.0	0
1.	.0	0	.0	0	.0	0	.0	0	.0	0	.0	0	20.0	1
2.	.0	0	.0	0	.0	0	.0	0	.0	0	.0	0	40.0	2
3.	100.0	1	100.0	0	100.0	0	100.0	0	100.0	0	100.0	0	100.0	2
4.	100.0	1	100.0	0	100.0	0	100.0	0	100.0	0	100.0	0	120.0	1
5.	100.0	1	100.0	0	100.0	0	100.0	0	100.0	0	100.0	0	140.0	2
6.	200.0	2	200.0	0	200.0	0	200.0	0	200.0	0	200.0	0	200.0	2
7.	200.0	2	200.0	0	435.0	1	435.0	0	435.0	0	435.0	0	435.0	2
8.	200.0	2	200.0	0	435.0	1	435.0	0	435.0	0	435.0	0	455.0	1
9.	300.0	3	450.0	1	450.0	0	480.0	1	480.0	0	480.0	0	480.0	0
10.	300.0	3	450.0	1	535.0	1	535.0	0	535.0	0	535.0	0	535.0	0
11.	300.0	3	450.0	1	535.0	1	535.0	0	535.0	0	535.0	0	555.0	1
12.	400.0	4	550.0	1	550.0	0	580.0	1	580.0	0	580.0	0	580.0	0
13.	400.0	4	550.0	1	635.0	1	635.0	0	635.0	0	635.0	0	635.0	0
14.	400.0	4	550.0	1	870.0	2	870.0	0	870.0	0	870.0	0	870.0	0
15.	500.0	5	650.0	1	870.0	2	870.0	0	870.0	0	870.0	0	890.0	1
16.	500.0	5	650.0	1	885.0	1	915.0	1	915.0	0	915.0	0	915.0	0
17.	500.0	5	650.0	1	970.0	2	970.0	0	970.0	0	970.0	0	970.0	0
18.	600.0	6	900.0	2	970.0	2	970.0	0	970.0	0	970.0	0	990.0	1
19.	600.0	6	900.0	2	985.0	2	985.0	1	1015.0	1	1015.0	0	1015.0	0
20.	600.0	6	900.0	2	1070.0	2	1070.0	2	1070.0	1	1070.0	0	1070.0	0
21.	700.0	7	1000.0	2	1305.0	3	1305.0	2	1305.0	1	1305.0	0	1305.0	0
22.	700.0	7	1000.0	2	1305.0	3	1305.0	2	1305.0	1	1305.0	0	1325.0	1
23.	700.0	7	1000.0	2	1320.0	3	1350.0	2	1350.0	1	1350.0	0	1350.0	0
24.	800.0	8	1100.0	2	1405.0	3	1405.0	2	1405.0	1	1405.0	0	1405.0	0
25.	800.0	8	1100.0	2	1405.0	3	1405.0	2	1405.0	1	1405.0	0	1425.0	1
26.	800.0	8	1100.0	2	1420.0	3	1450.0	2	1450.0	1	1450.0	0	1450.0	0
27.	900.0	9	1350.0	2	1505.0	3	1505.0	2	1505.0	1	1505.0	0	1505.0	0
28.	900.0	9	1350.0	2	1740.0	4	1740.0	2	1740.0	1	1740.0	0	1740.0	0
29.	900.0	9	1350.0	2	1740.0	4	1740.0	2	1740.0	1	1740.0	0	1760.0	1
30.	1000.0	10	1450.0	2	1755.0	4	1785.0	2	1785.0	1	1785.0	0	1785.0	0

Lista 8.3

Scanned with OKEN Scanner

TABELA REZULTATELOR:
****

CAP.TOT.	VALOARE	NUMAR UTILAJ 1	2	3	4	5	6	7
30.	1785.	0	0	3	1	0	0	0
28.	1740.	0	0	4	0	0	0	0
26.	1450.	1	0	2	1	0	0	0
24.	1405.	1	0	3	0	0	0	0
21.	1305.	0	0	3	0	0	0	0
20.	1070.	2	0	2	0	0	0	0

Lista 8.4

În acest caz, avem de maximizat funcţia:

$$R(x_1, x_2, \ldots, x_N) = g_1(x_1) + g_2(x_2) + \ldots + g_N(x_N),$$

supusă următoarelor restricţii:

$$x_i \geqslant 0,$$

$$\sum_{i=1}^{N} a_i(x_i) \leqslant x, \qquad (8-8)$$

$$\sum_{i=1}^{N} b_i(x_i) \leqslant y$$

Mergînd pe aceeaşi cale ca pînă acum, definim şirurile de funcţii:

$$f_N(x, y) = \max_{\{x\}} [g_1(x_1) + g_2(x_2) + \ldots + g_N(x_N)],$$

unde maximul se caută pe domeniul lui x dat de cele trei restricţii din (8—8).

Ecuaţia programării dinamice ia forma:

$$f_N(x, y) = \max_{\substack{a_N(x_N) \leqslant x \\ b_N(x_N) \leqslant y}} [g_N(x_N) + f_{N-1}(x - a_N(x_N), y - b_N(x_N)].$$

Pentru $N = 1$:

$$f_1(x, y) = \max g_1(x_1)$$
$$a_1(x_1) \leqslant x$$
$$b_1(x_1) \leqslant y$$

Un exemplu ne va arăta procedeul de lucru pentru utilizarea unui calculator.

Ex. 8—2. Să admitem că la problema rezolvată anterior (v. *Ex. 8—1*), de încărcare optimă a unui vapor cu utilaje petroliere de diferite greutăţi, pe lîngă restricţia de greutate totală P, ni se impune şi o restricţie de spaţiu total S.

Aceste restricţii sînt, firește, datorate capacităţii limitate — în greutate şi volum — pe care o are vasul.

Scanned with OKEN Scanner

Formulăm aşadar problema: „cîte articole de tipul i vom putea încărca pe vapor în condiţiile unor restricţii de greutate şi volum, în aşa fel ca valoarea totală a utilajelor încărcate să fie maximă?"

Vom folosi notaţiile:

v_i — valoarea celui de-al i-lea tip de utilaj;
p_i — greutatea celui de-al i-lea tip de utilaj;
s_i — spaţiul (volumul) celui de-al i-lea tip de utilaj.

Restricţiile, în cazul acestui exemplu, vor fi:

$$x_i \in \{0,\ 1,\ 2,\ \ldots\} \text{ (numere întregi)},$$

$$\sum_{i=1}^{N} x_i s_i \leqslant S,$$

$$\sum_{i=1}^{N} x_i p_i \leqslant P.$$

Valoarea încărcăturii totale este dată de funcţia:

$$R(x_1,\ x_2,\ \ldots,\ x_N) = \sum_{i=1}^{N} g_i(x_i),$$

unde ca şi în problema din *Ex. 8—1*:

$$g_i(x_i) = x_i v_i.$$

Relaţia fundamentală de recurenţă este aici:

$$f_k(s,\ p) = max\ [g_k(x_k) + f_{k-1}(p - x_k p_k,\ s - x_k s_k),$$

iar x_k poate lua valorile 0, 1, ..., $min\left\{\left[\frac{p}{p_k}\right], \left[\frac{s}{s_k}\right]\right\}$, unde $\left[\frac{p}{p_k}\right]$ sau $\left[\frac{s}{s_k}\right]$ înseamnă cel mai mare număr întreg pe care-l poate lua x (partea întreagă a lui x).

Rezolvarea problemei se realizează conform schemei de calcul din figura 8.4, prelucrarea cu un calculator automat efectuîndu-se cu ajutorul programului FORTRAN redat în lista 8.5 (asemănător celui din lista 8.1, folosit pentru problema din *Ex. 8—1*, dar la care se mai adaugă şi restricţia de volum). Folosind datele din lista 8.6 şi tabelele de lucru redate parţial în lista 8.7, obţinem rezultatele din „listing"-ul reprodus în lista 8.8.

*
* *

Vom examina acum o problemă de **programare dinamică neliniară,** adică un model de optimizare a unei funcţii neliniare, care poate apare, de exemplu, la evaluarea probabilităţii de bună funcţionare a unui dispozitiv electronic; acelaşi model poate fi aplicat şi unor grupuri de maşini sau utilaje care funcţionează în interdependenţă. Se pune condiţia ca dispozitivul electronic sau grupa de maşini să funcţioneze continuu. În acest scop se vor duplica unele componente şi se vor lega în paralel pentru ca,

Scanned with OKEN Scanner

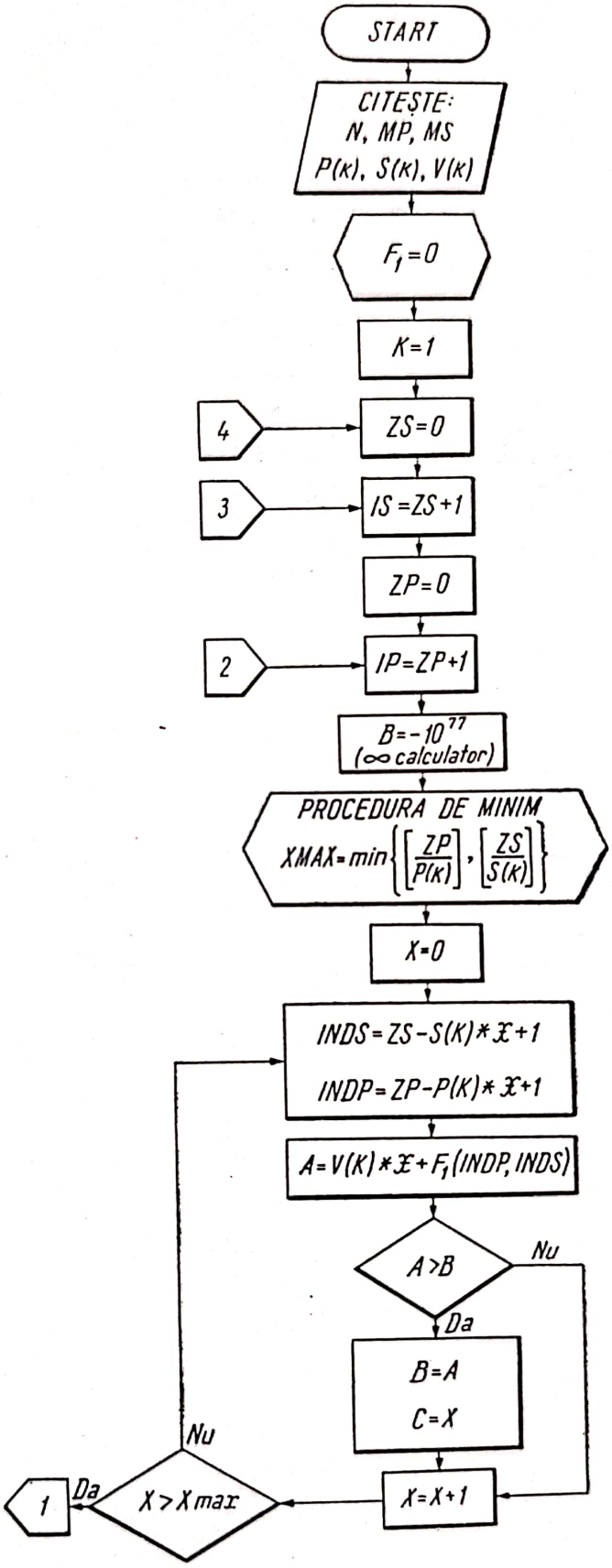
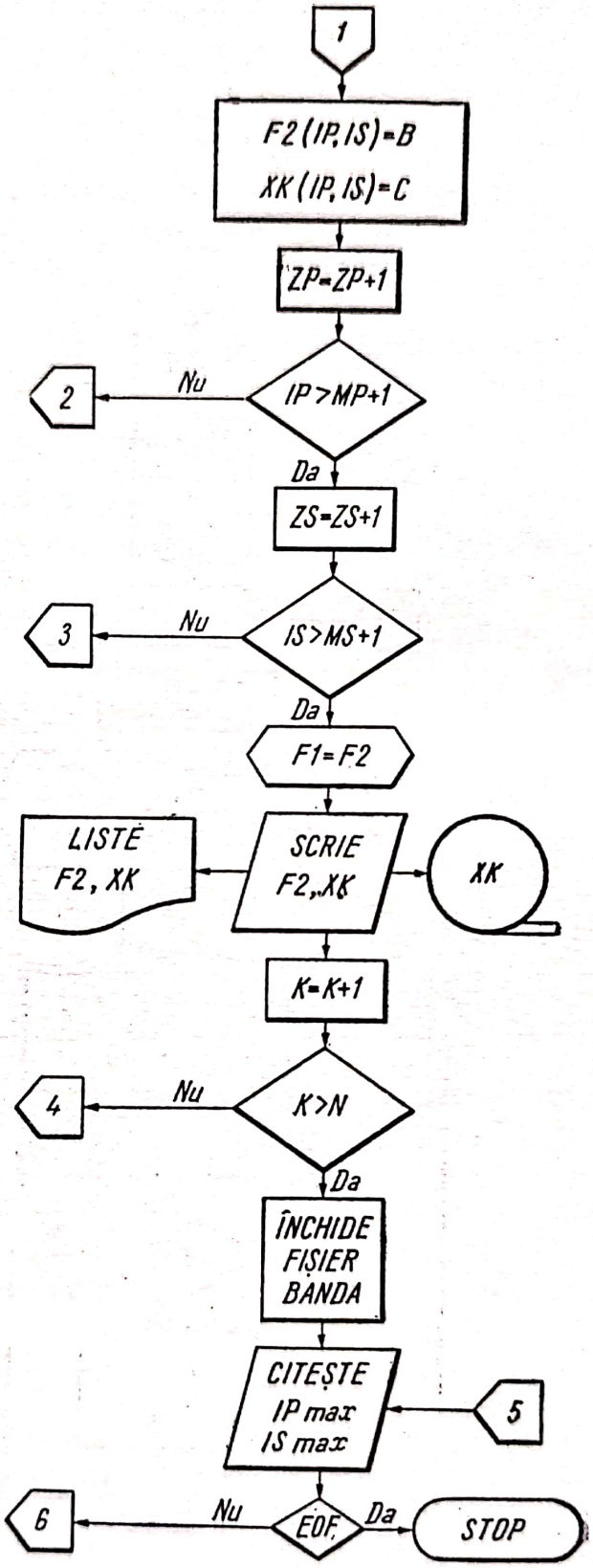
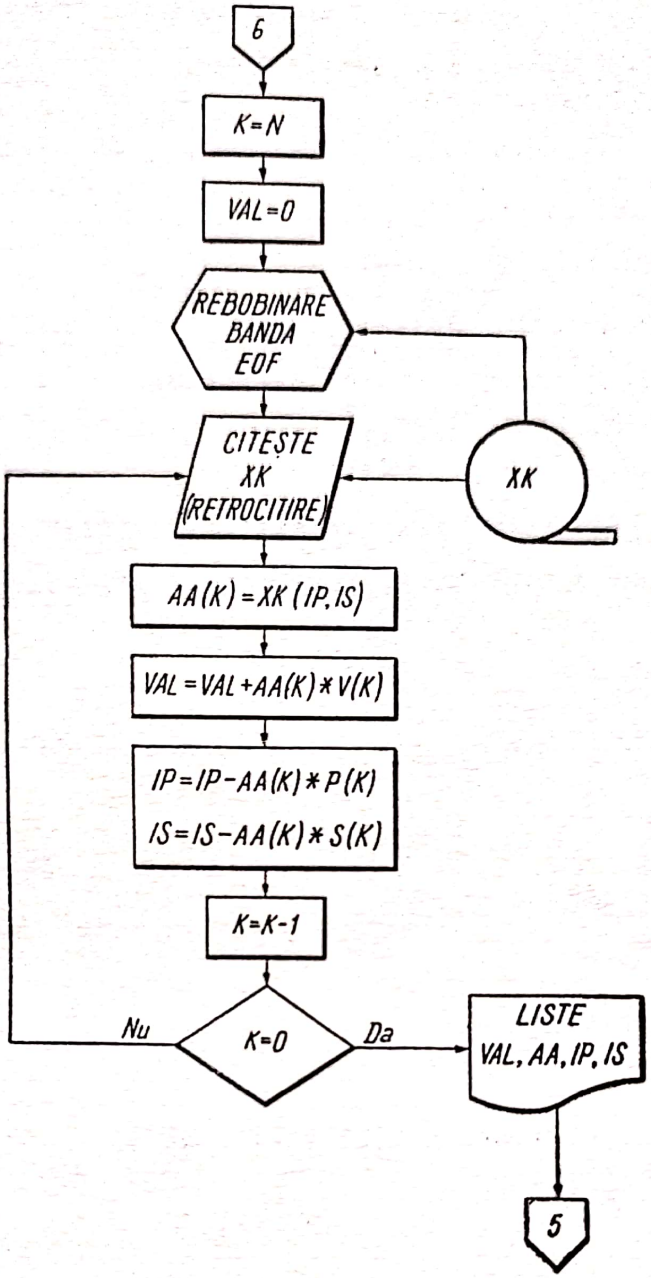


Fig. 8.4

Scanned with OKEN Scanner



Scanned with OKEN Scanner



în cazul uei defecțiuni la una din componente, sarcina ei să poată fi preluată de componentul paralel, în mod automat.

Din punct de vedere practic, nu putem dubla la nesfîrșit componentele din diferite considerente de spațiu, greutate, cost etc. Se pune întrebarea cum vom duplica în mod optim componentele, adică care este numărul din fiecare component care trebuie duplicat în așa fel încît probabilitatea de bună funcționare a întregului ansamblu să fie maximă.

Funcția de siguranță în funcționare a dispozitivului are următoarea expresie:

$$S_N = \prod_{j=1}^{N} G_j(m_j) \qquad m_i = 0,\ 1,\ 2,\ \ldots\ ,$$

unde $G_j(m_j)$ este funcția de bună funcționare pentru componentul j cînd se cunosc $1 + m_j$ componente similare legate în paralel.

Scanned with OKEN Scanner

```
C         LISTA 8,5
C         PROGRAMARE DINAMICA
C         PROGRAM BIDIMENSIONAL   PENTRU FUNCTIE DE N VARIABILE
C      (EX,8-2: INCARCARE UTILAJE CU RESTRICTII DE GREUTATE SI DE VOLUM
C
* DEFINE FILE 80(DVT:MT,RCF:V,BFS:659)=100
      IMPLICIT INTEGER (A-Z)
C
      DIMENSION P(8),S(8),V(8),F1(31,21),F2(31,21),XK(31,21),AA(8)
      DIMENSION TP(31),TS(21)
      DATA B8/Z880000000/
C
C         N     NUMARUL DE UTILAJE
C         MP    GREUTATE TOTALA MAXIMA
C         MS    VOLUM TOTAL MAXIM
C         P,S,V  GREUTATILE,VOLUMELE SI VALOAREA FIECARUI UTILAJ
C         B8    MINUS INFINIT PENTRU CALCULATOR
C         K     INDICELE UTILAJULUI
C         IP    INDICELE GREUTATII
C         IS    INDICELE VOLUMULUI
C
      READ (105,200) N,MP,MS,(P(I),S(I),V(I),I=1,N)
C
      WRITE(108,400) N,MP,MS,(I,P(I),S(I),V(I),I=1,N)
      NP=MP+1
      NS=MS+1
      DO 1 I=1,NP
      TP(I)=I-1
      DO 1 J=1,NS
    1 F1(I,J)=0
C
      DO 9 K=1,N
      ZS=0
    2 IS=ZS+1
      ZP=0
    3 IP=ZP+1
      B=B8
C
C     PROCEDURA DE MINIM
      I=0
      XMAX=ZP
      YMAX=ZS
    4 IF ( XMAX.LT.P(K).OR.YMAX.LT.S(K) ) GO TO 5
      XMAX=XMAX-P(K)
      YMAX=YMAX-S(K)
      I=I+1
      GO TO 4
C
    5 XMAX=I
      X=0
    6 INDP=ZP-P(K)*X+1
      INDS=ZS-S(K)*X+1
      A=V(K)*X+F1(INDP,INDS)
      IF(A.LE.B) GO TO 7
      B=A
      C=X
    7 X=X+1
      IF (X.LE.XMAX) GO TO 6
C
C     RETINEREA VALORII MAXIME
      F2(IP,IS)=B
      XK(IP,IS)=C
      ZP=ZP+1
      IF (IP.LT.NP) GO TO 3
      ZS=ZS+1
      IF (IS.LT.NS) GO TO 2
```


Scanned with OKEN Scanner

```
      DO 8 I=1,NS
      TS(I)=I-1
      DO 8 J=1,NP
    8 F1(J,I)=F2(J,I)
C
      WRITE (108,401) K,(TS(I),I=1,NS),(TP(I),(F1(I,J),J=1,NS),I=1,NP)
      WRITE (108,402) K,(TS(I),I=1,NS),(TP(I),(XK(I,J),J=1,NS),I=1,NP)
    9 WRITE (100) XK
      ENDFILE 100
      WRITE (108,403) (I,I=1,N)
C        EXPLORAREA TABELELOR
   10 READ (105,200,END=14) MAXP,MAXS
   11 READ (100,END=12)
      GO TO 11
   12 CONTINUE
      IP=MAXP+1
      IS=MAXS+1
      VAL=0
      K=N
      BACKSPACE 100
   13 BACKSPACE 100
      READ(100) XK
      BACKSPACE 100
      AA(K)=XK(IP,IS)
      VAL=VAL+AA(K)*V(K)
      IP=IP-AA(K)*P(K)
      IS=IS-AA(K)*S(K)
      K=K-1
      IF (K.GT.0) GO TO 13
      IP=IP-1
      IS=IS-1
      WRITE (108,404) VAL,MAXP,MAXS,IP,IS,(AA(I),I=1,N)
      GO TO 10
   14 STOP
C**********
  200 FORMAT ( 5I10)
  400 FORMAT ('1       DATELE DE INTRARE:'/6X,5(4H****)//
     1          '     NUMAR UTILAJE    ',I12 /
     2          '     GREUTATE MAXIMA  ',I12 /
     3          '     VOLUM MAXIM      ',I12 //
     4          '     N,UT.  GREUTATE  VOLUM  VALOARE '/(4I8))
  401 FORMAT ('1',T54,'EXTRAS DIN TABELELE DE LUCRU:'/T54,30(1H*)/
     *T66,'F',I2//'    P/S',21I6//(22I6))
  402 FORMAT ('1',T54,'EXTRAS DIN TABELELE DE LUCRU:'/T54,30(1H*)/
     *T66,'X',I2//'    P/S',21I6//(22I6))
  403 FORMAT ('1',35X,'TABELA REZULTATELOR:'/35X,22(1H*)/
     *'0   VALOARE  GR. MAX  VOL.MAX  REST GR  REST  VOL',8X,
     *'N U M A R  U T I L A J'/ 49X,7I5/)
  404 FORMAT (5I9,4X,7I5)
C**********
      END
```

Lista 8.5

```
      DATELE DE INTRARE:
      ******************
NUMAR UTILAJE                    7
GREUTATE MAXIMA                 30
VOLUM MAXIM                     20

N,UT.   GREUTATE   VOLUM   VALOARE
    1         3        3       100
    2         9       10       650
    3         7        4       435
    4         9        5       480
    5         8        4       435
    6         7        3       425
    7         1        1        20
```

Lista 8.6

Scanned with OKEN Scanner

Lista 8.7

EXTRAS DIN TABELELE DE LUCRU:
****
F 6

P/S	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
5	0	0	0	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100	100
6	0	0	0	100	100	100	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
7	0	0	0	425	435	435	435	435	435	435	435	435	435	435	435	435	435	435	435	435	435
8	0	0	0	425	435	435	435	435	435	435	435	435	435	435	435	435	435	435	435	435	435
9	0	0	0	425	435	480	480	480	480	480	480	480	480	480	480	480	480	480	480	480	480
10	0	0	0	425	435	480	525	535	535	535	535	535	535	535	535	535	535	535	535	535	535
11	0	0	0	425	435	480	525	535	535	535	535	535	535	535	535	535	535	535	535	535	535
12	0	0	0	425	435	480	525	535	580	580	580	580	580	580	580	580	580	580	580	580	580
13	0	0	0	425	435	480	525	535	580	625	635	635	635	635	635	635	635	635	635	635	635
14	0	0	0	425	435	480	850	860	870	870	870	870	870	870	870	870	870	870	870	870	870
15	0	0	0	425	435	480	850	860	870	870	870	870	870	870	870	870	870	870	870	870	870
16	0	0	0	425	435	480	850	860	905	915	915	915	915	915	915	915	915	915	915	915	915
17	0	0	0	425	435	480	850	860	905	950	960	970	970	970	970	970	970	970	970	970	970
18	0	0	0	425	435	480	850	860	905	950	960	970	970	970	970	970	970	970	970	970	970
19	0	0	0	425	435	480	850	860	905	950	960	1005	1015	1015	1015	1015	1015	1015	1015	1015	1015
20	0	0	0	425	435	480	850	860	905	950	960	1005	1050	1060	1070	1070	1070	1070	1070	1070	1070
21	0	0	0	425	435	480	850	860	905	1275	1285	1295	1305	1305	1305	1305	1305	1305	1305	1305	1305
22	0	0	0	425	435	480	850	860	905	1275	1285	1295	1305	1305	1305	1305	1305	1305	1305	1305	1305
23	0	0	0	425	435	480	850	860	905	1275	1285	1330	1340	1350	1350	1350	1350	1350	1350	1350	1350
24	0	0	0	425	435	480	850	860	905	1275	1285	1330	1375	1385	1395	1405	1405	1405	1405	1405	1405
25	0	0	0	425	435	480	850	860	905	1275	1285	1330	1375	1385	1395	1405	1405	1405	1405	1405	1405
26	0	0	0	425	435	480	850	860	905	1275	1285	1330	1375	1385	1430	1440	1450	1450	1450	1450	1450
27	0	0	0	425	435	480	850	860	905	1275	1285	1330	1375	1385	1430	1475	1485	1495	1505	1505	1505
28	0	0	0	425	435	480	850	860	905	1275	1285	1330	1700	1710	1720	1730	1740	1740	1740	1740	1740
29	0	0	0	425	435	480	850	860	905	1275	1285	1330	1700	1710	1720	1730	1740	1740	1740	1740	1740
30	0	0	0	425	435	480	850	860	905	1275	1285	1330	1700	1710	1755	1765	1775	1785	1785	1785	1785

Scanned with OKEN Scanner

Lista 8.7 (continuare)

EXTRAS DIN TABELELE DE LUCRU:
****
X 6

P/S	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0	0	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	0	0	0	1	2	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	0	0	0	1	2	3	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
5	0	0	0	1	2	3	4	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6	0	0	0	1	2	3	4	5	6	6	6	6	6	6	6	6	6	6	6	6	6
7	0	0	0	1	2	3	4	5	6	7	7	7	7	7	7	7	7	7	7	7	7
8	0	0	0	1	2	3	4	5	6	7	8	8	8	8	8	8	8	8	8	8	8
9	0	0	0	1	2	3	4	5	6	7	8	9	9	9	9	9	9	9	9	9	9
10	0	0	0	1	2	3	4	5	6	7	8	9	10	10	10	10	10	10	10	10	10
11	0	0	0	1	2	3	4	5	6	7	8	9	10	11	11	11	11	11	11	11	11
12	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	12	12	12	12	12	12
13	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	13	13	13	13	13
14	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	14	14	14	14
15	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	15	15	15
16	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	16	16
17	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	17
18	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
19	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
20	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
21	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
22	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
23	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
24	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
25	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
26	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
27	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
28	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
29	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
30	0	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

Scanned with OKEN Scanner

TABELA REZULTATELOR:

VALOARE	GR. MAX	VOL.MAX	REST GR	REST VOL	NUMAR UTILAJ 1	2	3	4	5	6	7
1785	30	20	0	3	0	0	3	1	0	0	0
1760	29	19	0	2	0	0	1	0	0	0	1
1740	28	18	0	2	0	0	4	0	0	0	0
1495	27	17	0	0	2	0	2	0	0	1	0
1450	26	16	0	0	1	0	2	1	0	0	0
1410	25	15	0	0	1	0	2	0	0	1	1
1395	24	14	0	0	1	0	2	0	0	1	0
1330	22	12	0	0	0	0	2	0	0	1	1
1295	21	11	0	0	0	0	2	0	0	1	0
970	20	10	2	0	1	0	0	0	0	2	1
485	10	5	1	0	0	0	0	1	0	0	0

Lista 8.8

Forma funcţiilor $g_j(m_j)$ este tipică pentru astfel de probleme:

$$G_j(m_j) = 1 - (1 - p_j)^{1+m_j},$$

unde p_j este probabilitatea de bună funcţionare a fiecărui component j, cunoscută.

Asupra restricţiilor pe care trebuie să le introducem vom face o analiză apriorică pentru a evita un efort de calcul mult prea mare; ştim că fiecare restricţie conduce la mărirea cu o dimensiune a tabelelor de lucru.

Să presupunem, deci, că în urma analizei făcute am reuşit să stabilim pentru fiecare component numărul maxim de duplicate. Această analiză s-a făcut ţinînd cont de restricţiile de greutate, de spaţiu disponibil, de cost, de posibilitate de conectare în caz de defecţiune etc. Tot în urma analizei, s-a stabilit şi numărul maxim total de componente în dublură pe care îl poate conţine tot dispozitivul; fie acesta MCM.

Problema constă în aceea a maximizării funcţiei:

$$S_N = \prod_{j=1}^{N} G_j(m_j),$$

supusă restricţiilor:

$$\sum_{j=1}^{N} m_j \leqslant MCM, \qquad (8-9)$$

$$m_j \geqslant 0.$$

Ecuaţia de recurenţă a programării dinamice are pentru această funcţie forma:

$$F_N(m) = \max_{m_N \in [0, Mx(m)]} [G_N(m_N) \cdot F_{N-1}(m - m_n)],$$

cu:

$$F_1(m) = \max_{m_1 \in (0, Mx(1))} G_1(m_1),$$

Scanned with OKEN Scanner

unde tabela M_x cuprinde pentru fiecare tip de component (j) numărul maxim de dubluri.

Ex. 8—3. Pentru ilustrarea celor de mai sus, vom considera exemplul unui grup de agregate cu 6 componente, la care numărul total de „dubluri" poate ajunge pînă la 40.

Rezolvarea problemei se poate efectua cu programul de calcul redat în lista 8.9, redactat în FORTRAN.

Problema a fost rezolvată pentru datele indicate în lista 8.10. Este interesant de urmărit în tabela cu rezultate (lista 8.11) evoluţia numărului de componente în funcţie de probabilitatea de bună funcţionare şi de suma lor totală.

TABELELE DE LUCRU FK(X) SI XK(X):

M	F 1	X 1	F 2	X 2	F 3	X 3	F 4	X 4	F 5	X 5	F 6	X 6
0	.880	0	.616	0	.585	0	.439	0	.351	0	.298	0
1	.986	1	.801	1	.761	0	.571	0	.456	0	.388	0
2	.998	2	.897	1	.852	0	.713	1	.571	0	.485	0
3	1.000	3	.959	2	.911	0	.799	1	.685	1	.582	0
4	1.000	4	.978	3	.957	1	.854	1	.767	1	.669	1
5	1.000	5	.990	3	.975	1	.897	1	.820	1	.750	1
6	1.000	6	.996	4	.988	1	.942	2	.861	1	.801	1
7	1.000	7	.998	5	.993	1	.960	2	.904	1	.842	1
8	1.000	7	.999	6	.996	2	.972	2	.934	2	.884	1
9	1.000	7	1.000	6	.997	2	.984	3	.952	2	.913	1
10	1.000	7	1.000	6	.999	2	.989	3	.964	2	.931	2
11	1.000	7	1.000	7	.999	2	.992	4	.976	2	.949	2
12	1.000	7	1.000	8	1.000	2	.995	4	.982	3	.961	2
13	1.000	7	1.000	8	1.000	2	.996	4	.988	3	.973	2
14	1.000	7	1.000	9	1.000	3	.998	4	.991	3	.979	2
15	1.000	7	1.000	10	1.000	3	.999	5	.993	3	.985	2
16	1.000	7	1.000	10	1.000	3	.999	5	.995	3	.987	2
17	1.000	7	1.000	10	1.000	3	.999	5	.996	3	.990	3
18	1.000	7	1.000	10	1.000	4	1.000	5	.998	4	.993	3
19	1.000	7	1.000	10	1.000	4	1.000	5	.998	4	.994	3
20	1.000	7	1.000	10	1.000	4	1.000	5	.999	4	.996	3
21	1.000	7	1.000	10	1.000	4	1.000	5	.999	5	.997	3
22	1.000	7	1.000	10	1.000	4	1.000	5	.999	5	.998	3
23	1.000	7	1.000	10	1.000	4	1.000	5	.999	5	.998	3
24	1.000	7	1.000	10	1.000	4	1.000	5	1.000	5	.999	4
25	1.000	7	1.000	10	1.000	4	1.000	5	1.000	6	.999	4
26	1.000	7	1.000	10	1.000	4	1.000	5	1.000	6	.999	4
27	1.000	7	1.000	10	1.000	4	1.000	5	1.000	6	.999	4
28	1.000	7	1.000	10	1.000	4	1.000	5	1.000	6	1.000	4
29	1.000	7	1.000	10	1.000	4	1.000	5	1.000	7	1.000	5
30	1.000	7	1.000	10	1.000	4	1.000	5	1.000	7	1.000	5
31	1.000	7	1.000	10	1.000	4	1.000	5	1.000	7	1.000	5
32	1.000	7	1.000	10	1.000	4	1.000	5	1.000	7	1.000	5
33	1.000	7	1.000	10	1.000	4	1.000	5	1.000	8	1.000	5
34	1.000	7	1.000	10	1.000	4	1.000	5	1.000	8	1.000	5
35	1.000	7	1.000	10	1.000	4	1.000	5	1.000	9	1.000	6
36	1.000	7	1.000	10	1.000	4	1.000	5	1.000	10	1.000	6
37	1.000	7	1.000	10	1.000	4	1.000	5	1.000	10	1.000	6
38	1.000	7	1.000	10	1.000	4	1.000	5	1.000	10	1.000	6
39	1.000	7	1.000	10	1.000	4	1.000	5	1.000	10	1.000	6
40	1.000	7	1.000	10	1.000	4	1.000	5	1.000	10	1.000	7

Scanned with OKEN Scanner

```
C       LISTA 8,9
C       PROGRAMARE DINAMICA
C     (EX.8-9: MODEL DE ASAMBLARE CU SIGURANTA IN FUNCT. COMPONENTELOR)
C
C           NUMARUL DE COMPONENTE DIFERITE          N
C           NUMARUL MAXIM DE COMP.IN MONTAJ         MX
C           PROBABILITATEA DE BUNA FUNCTIONARE      P
C           NUMARUL MAXIM TOTAL AL COMPONENTELOR    MCM
C
      DIMENSION MX(6),P(6),F1(41),F2(41),XK(41),TABEL(41,20),AA(6)
      DATA B8/Z880000000/
C
      G(M,PROB)=1-(1-PROB)**(M+1)
C
      READ (105,200) N,MCM,(MX(I),P(I),I=1,N)
      WRITE(108,400) N,MCM,(I,MX(I),P(I),I=1,N)
      WRITE(108,401) (I,I,I=1,N)
      MCM1=MCM+1
      DO 1 I=1,MCM1
    1 F1(I)=1
      DO 5 K=1,N
      MZ=0
    2 B=B8
      MMAX=MX(K)
      IF (MX(K).GT.MZ) MMAX=MZ
      M=0
    3 IND=MZ-M+1
      A=G(M,P(K))*F1(IND)
      IF (A.LE.B) GO TO 4
      B=A
      C=M
    4 M=M+1
      IF (M.LE.MMAX) GO TO 3
      F2 (MZ+1)=B
      XK (MZ+1)=C
      MZ=MZ+1
      IF (MZ.LE.MCM) GO TO 2
      DO 5 I=1,MCM1
      TABEL (I,2*K-1)=F2(I)
      TABEL (I,2*K  )=XK(I)
    5 F1(I)=F2(I)
C
      N2=N+N
      DO 6 I=1,MCM1
      M=I-1
    6 WRITE (108,402) M,(TABEL(I,J),J=1,N2)
C
C     EXPLORARE TABELE
C
      WRITE(108,403)(I,I=1,N)
      DO 8 I=0,30,2
      K=N
      IZ=MCM1-I
    7 AA(K)=TABEL (IZ,2*K)
      IZ=IZ-AA(K)
      K=K-1
      IF (K.GT.0) GO TO 7
      MCM=MCM1-1-I
      WRITE (108,404) MCM,(AA(K),K=1,N)
    8 CONTINUE
      STOP
```


Scanned with OKEN Scanner

```
C***********
  200 FORMAT (I10,F10.2)
  400 FORMAT ('1      DATELE DE INTRARE:'/6X,5(4H****)/
     *'0    COMPONENTE',I10/'      NUMAR TOTAL',I9//5X,'N      NR.MAX',
     *5X,'PROB.'/(I6,I9,F10.2))
  401 FORMAT ('1',22X,'TABELELE DE LUCRU FK(X) SI XK(X):'/22X,35(1H*)/
     *'0   M ',6('   F',I2,'   X',I2))
  402 FORMAT(I6,  10(F6.3,I6))
  403 FORMAT('1',15X,'TABELA REZULTATELOR:'/15X,22(1H*)/
     *'0 TOTAL COMP     N U M A R   C O M P O N E N T'/14X,6I5/)
  404 FORMAT (I8,6X, 6I5)
C***********
      END
```

Lista 8.9

DATELE DE INTRARE:
****

COMPONENTE 6
NUMAR TOTAL 40

N	NR.MAX	PROB.
1	8	.88
2	10	.70
3	4	.95
4	5	.75
5	10	.80
6	8	.85

Lista 8.10

TABELA REZULTATELOR:
****

TOTAL COMP.	NUMAR COMPONENT					
	1	2	3	4	5	6
40	6	10	4	5	8	7
38	6	10	4	5	7	6
36	5	9	4	5	7	6
34	5	9	3	5	7	5
32	5	8	3	5	6	5
30	4	7	3	5	6	5
28	4	7	3	5	5	4
26	4	6	2	5	5	4
24	3	6	2	5	4	4
22	3	5	2	5	4	3
20	3	5	2	4	3	3
18	2	4	2	4	3	2
16	2	4	1	4	3	2
14	2	3	1	3	3	2
12	2	3	1	2	2	2
10	1	2	1	2	2	2

Lista 8.11

Scanned with OKEN Scanner

8.3. PRINCIPIILE MODELĂRII DINAMICE

Optimizarea sistemelor* prin *modelare dinamică* constă în reprezentarea matematică a relaţiilor din sistemul considerat, deci în elaborarea unui *model* în care apar probleme de extremum, asupra căruia se efectuează apoi un şir de *simulări* prin aplicarea unor algoritmi de optimizare. Sistemul dat se mai numeşte şi sistem original, sau sistem obiect, sau — încă — *sistem real*, iar modelul care îl reprezintă şi asupra căruia se aplică simularea se mai numeşte, de aceea, şi *model de simulare*. Prin această simulare se caută un set de valori pentru parametrii modelului, pe baza căruia să se realizeze conducerea în timp real a sistemului original, de tipul unui reglaj automat.

Comanda automată a unui sistem se realizează după cum ştim, pe baza unei scheme de principiu de forma celei reprezentate în figura 8.5. Pe desenul din această figură am notat cu litera R corpul de comandă, iar cu litera S corpul comandat, fie el un sistem real oarecare. Cu $x(t)$ am notat starea sistemului la un moment dat, reprezentabilă prin parametrii de ieşire $x_1(t)$, $x_2(t)$, ..., $x_n(t)$ care conţin informaţia necesară pentru determinarea comportării sistemului la fiecare moment de timp t. Aceşti parametri sînt coordonatele unui vector:

$$\mathbf{x}(t) = \{x_1(t),\ x_2(t),\ \ldots,\ x_n(t)\}, \tag{8—10}$$

numit *vectorul de stare* al sistemului.

Corpul de comandă R transmite corpului condus S o comandă $u(t)$. Mai multe comenzi $u_1(t)$, $u_2(t)$, ..., $u_m(t)$ pot fi considerate coordonatele unui vector:

$$\mathbf{u}(t) = \{u_1(t),\ u_2(t),\ \ldots,\ u_m(t)\}, \tag{8—11}$$

numit *vectorul comandă*.

La intrare, corpul de comandă primeşte un semnal $x^*(t)$ care reprezintă *scopul comenzii*, adică defineşte mărimile de stare care sînt prevăzute pentru corpul comandat S. Fie acestea coordonatele vectorului:

$$x^*(t) = \{x_1^*(t),\ x_2^*(t),\ \ldots,\ x_k^*(t)\}. \tag{8—12}$$

După cum ştim, sistemele de comandă automată se împart în două clase: sistemele cu buclă deschisă şi sistemele cu buclă închisă; acestea din urmă se mai numesc şi sisteme cu buclă de reacţie.

În sistemele cu buclă deschisă, corpul de comandă nu primeşte informaţie asupra stării reale a corpului comandat; în sistemele cu buclă închisă corpul de comandă primeşte informaţii asupra stării $x(t)$ a corpului comandat, iar dacă aceasta nu corespunde cu starea prevăzută $x^*(t)$, corpul de comandă

---

* Prin sistem vom înţelege, în accepţiunea largă a cuvîntului, o colecţie de obiecte în interacţiune şi integrate în scopul realizării unui anumit obiectiv. În această definiţie este esenţială ideea de obiectiv, sistem fiind numai acea colecţie de elemente capabile de interacţiune într-un astfel de mod încît să fie realizat un program dat, de unde şi necesitatea elaborării unor procedee de conducere optimală.

Scanned with OKEN Scanner

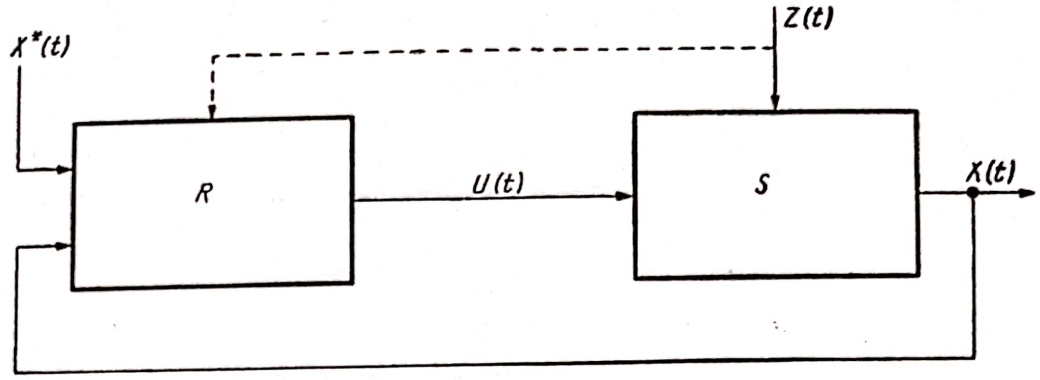


Fig. 8.5

emite o comandă $u(t)$ care are ca scop să-l apropie pe $x(t)$ de $x^*(t)$. Ecartul între cele două mărimi poate fi datorit fie întîrzierii cu care se primeşte informaţia asupra stării sistemului comandat, fie posibilităţilor limitate de comandă ale corpului R, fie, în fine, unei perturbaţii $z(t)$ imprevizibilă sau incontrolabilă, care acţionează asupra corpului S şi se reflectă în starea sa.

Perturbaţia $z(t)$ poate fi descrisă, de asemenea, de coordonatele unui vector:

$$\mathbf{z}(t) = \{z_1(t),\ z_2(t),\ \ldots,\ z_l(t)\}. \tag{8—13}$$

Dacă perturbaţia a putut fi măsurată, iar rezultatul măsurătorilor a fost canalizat spre corpul de comandă R, acesta poate produce o comandă $u(t + \Delta t)$, care să compenseze perturbaţia $z(t)$.

Metoda compensaţiei reunită cu metoda buclei de reacţie prezintă numeroase avantaje: în toate cazurile dezideratul este de a avea un corp de comandă care să conducă „cît mai bine" corpul comandat.

Pentru corpul comandat S nu se cer caracteristici deosebite, acestea fiind impuse de obicei corpului de comandă R, care trebuie să îndeplinească următoarele trei proprietăţi fundamentale:

1° să reflecte caracteristicile corpului S;

2° să aibă precizat scopul comenzii — mărimea $x^*(t)$;

3° să fie permanent în posesia informaţiei referitoare la starea corpului comandat S — mărimea $x(t)$.

Cerinţa „să reflecte caracteristicile corpului S" înseamnă definirea unei relaţii între mărimile $\mathbf{x}$ — de stare, $\mathbf{u}$ — de comandă, şi $\mathbf{z}$ — de perturbaţie. În acest sens, putem scrie într-un mod general:

$$\mathbf{x}(t) = \mathbf{F}(\mathbf{u}(t),\ \mathbf{z}(t)), \tag{8—14}$$

cu vectorii $\mathbf{x}(t)$, $\mathbf{u}(t)$ şi $\mathbf{z}(t)$ definiţi în spaţiul real R.

$\mathbf{F}$ din (8—14) este un operator, adică o lege de corespondenţă între două mulţimi de funcţii. Operatorul $\mathbf{F}$ poate fi definit în mai multe moduri: prin formule, prin grafice — respectiv tabele (funcţii tabelate), sau adesea printr-un sistem de ecuaţii diferenţiale.

Scanned with OKEN Scanner

În acest ultim caz putem scrie într-un mod condensat că:

$$\frac{dx}{dt} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{z}, t), \tag{8—15}$$

unde: $\mathbf{x} \in R^n$, $\mathbf{u} \in R^m$, $\mathbf{z} \in R^l$, iar t este un scalar.

Dacă fixăm vectorul de stare la momentul inițial t_0 (condițiile inițiale):

$$\mathbf{x}^0 = (\mathbf{x}_1^0, \mathbf{x}_2^0, \ldots, \mathbf{x}_n^0) \tag{8—16}$$

și dacă cunoaștem la fiecare moment de timp valorile comenzilor $\mathbf{u}(t)$ și ale perturbației $\mathbf{z}(t)$, atunci putem determina vectorul de stare $\mathbf{x}(t)$, adică putem afla soluția sistemului (8—15). Aceasta reprezintă o traiectorie în spațiul n-dimensional al stărilor sistemului.

Într-un mod general, ținînd cont de caracteristicile lor fizice, sistemele sînt privite ca sisteme continue, sisteme continue cu stări discrete și sisteme discrete. Reprezentarea lor poate fi făcută prin două mari clase de ecuații diferențiale: continue și cu diferențe finite.

Vom analiza aspectul sistemelor continue cu stări discrete pe care le reprezentăm prin ecuațiile cu diferențe finite.

Astfel, dacă la un moment de timp k vectorul de stare este:

$$\mathbf{x}(k) = \{x_1(k), x_2(k), \ldots, x_n(k)\}, \tag{8—17}$$

vectorul de comandă este:

$$\mathbf{u}(k) = \{u_1(k), u_2(k), \ldots, u_m(k)\} \tag{8—18}$$

și vectorul perturbație este:

$$\mathbf{z}(k) = \{z_1(k), z_2(k), \ldots, z_l(k)\}, \tag{8—19}$$

cu vectorii $\mathbf{x}(t)$, $\mathbf{u}(t)$, $\mathbf{z}(t)$ definiți în spațiul real R ca mai înainte, atunci ecuațiile cu diferențe finite pot fi scrise în forma:

$$x_i(k+1) = g_i(\mathbf{x}(k), \mathbf{u}(k), \mathbf{z}(k), k). \tag{8—20}$$
$$i = 1, 2, \ldots, n$$

Putem pune în evidență legătura între starea sistemului la două momente de timp vecine scriind:

$$x_i(k+1) = x_i(k) + \Delta t \cdot f_i(\mathbf{x}(k), \mathbf{u}(k), \mathbf{z}(k), k), \tag{8—21}$$
$$i = 1, 2, \ldots, n$$

unde Δt este intervalul de timp, ales cu teorema eșantionării, pentru calculul sistemului. Funcțiile f_i sînt în general funcții neliniare.

În fond, mai sus, avem de integrat un sistem de ecuații diferențiale după metoda liniei poligonale a lui Euler. Vom alege vectorul $\mathbf{x}^0$ la momentul inițial $k = 0$, vom fixa intervalul Δt și vom calcula cu ajutorul relației (8—21) la fiecare moment de timp k, $k + \Delta t$, $k + 2\Delta t$ etc., valorile variabilelor de stare $\mathbf{x}(k)$. Ne vom opri la un moment de timp $k = N$ cînd vom considera că valorile variabilelor de stare obținute și reținute pe parcursul procesului de calcul, definesc în mod concludent funcțiile f_i căutate.

Scanned with OKEN Scanner

Dacă corpul S reprezintă un sistem fizic, atunci ne putem aştepta ca anumite mărimi ale sale să varieze numai între anumite limite. În acest caz sistemului de ecuaţii (8—21) îi vom adăuga un set de restricţii.

Dacă componentele vectorului de comandă $\mathbf{u}$ pot avea numai valori cuprinse pe un anumit domeniu, vom scrie simbolic:

$$\mathbf{u} \in \Omega(\mathbf{u}). \tag{8—22}$$

Această formulare revine în multe cazuri la restricţii simple de forma:

$$u_i(t) \in [a_i, b_i] \quad i = 1, 2, \ldots, m. \tag{8—23}$$

Putem spune, în acest caz, că vectorul $\mathbf{u}$ este mărginit într-un paralelipiped de dimensiune m.

Tot la fel pot exista restricţii pentru vectorul $\mathbf{x}$ al stărilor. Vom spune în mod analog că în spaţiul lui x de dimensiune n, vectorul $\mathbf{x}$ este mărginit pe un anumit domeniu $\Omega(\mathbf{x})$ şi vom scrie simbolic:

$$\mathbf{x} \in \Omega(\mathbf{x}). \tag{8—24}$$

Într-un caz general, aceste „constrîngeri" pot fi impuse unei funcţionale depinzînd de $\mathbf{x}(t)$, $\mathbf{u}(t)$ şi $\mathbf{z}(t)$:

$$Q_\mu[\mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t), t] \in \Omega_\mu(Q). \tag{8—25}$$

A doua mare caracteristică impusă corpului de comandă este cea referitoare la scopul comenzii $\mathbf{x}^*(t)$. În cazul general, putem considera că avem de maximizat sau de minimizat o anumită mărime I, denumită *criteriu de optim* (sau *funcţie obiectiv*, sau încă *funcţională de performanţă* etc.) depinzînd de toate mărimile care au intervenit în sistem:

$$I(\mathbf{x}(t), \mathbf{x}^*(t), \mathbf{u}(t), \mathbf{z}(t), t) = min. \tag{8—26}$$

Această formulare generală poate fi, în cazuri concrete, mai redusă. Astfel criteriul de optim poate lua forma unei integrale atunci cînd se cunosc condiţiile iniţiale:

$$I = \int_0^t G(\mathbf{x}(t), \mathbf{u}(t), t)\mathrm{d}t \tag{8—27}$$

sau

$$I = \int_0^{t_f} H(\mathbf{x}(t_f), t)\mathrm{d}t, \tag{8—28}$$

unde t_f este timpul final.

În unele cazuri se cere minimizarea erorii medii patratice:

$$I = \int_0^t [\mathbf{x}(t) - \mathbf{x}^*(t)]^2 \, \mathrm{d}t,$$

Scanned with OKEN Scanner

deoarece în buclele de reacţie simple scopul comenzii este de a aduce starea $\mathbf{x}(t)$, la o anumită valoare $\mathbf{x}^*(t)$, impusă, la fiecare moment de timp.

A treia mare caracteristică impusă corpului de comandă este aceea de a fi, la fiecare moment de timp, în posesia valorilor vectorilor corpului comandat; altfel spus, corpul de comandă şi corpul comandat trebuie să fie legaţi prin bucla de reacţie (v. fig. 8.5).

Dacă corpul de comandă ar putea fi la fiecare moment de timp în posesia valorilor tuturor vectorilor, care de fapt definesc sistemul, adică dacă ar fi în posesia vectorului de stare $\mathbf{x}(t)$, a vectorului de comandă $\mathbf{u}(t)$ şi a vectorului de stare prevăzută $\mathbf{x}^*(t)$, dacă ar cunoaşte şi vectorul perturbaţie $\mathbf{z}(t)$, iar operatorul $\bar{\mathbf{F}}$ i-ar fi definit, atunci corpul de comandă ar fi în posesia unei informări complete asupra corpului comandat. Un asemenea sistem ipotetic ar putea fi prevăzut cu un program de comandă prestabilit: ar fi un sistem cu bucla deschisă şi nu ar mai fi nevoie de o „teorie a comenzii", corpul condus fiind „predestinat" să execute programul corpului de comandă.

În realitate nu avem un asemenea corp de comandă. Toate corpurile de comandă cunoscute sînt de fapt în posesia unei informări incomplete asupra corpului condus. Vor lipsi, pe rînd sau deodată, informaţii privitoare fie la vectorul perturbaţie (cel mai adesea), fie la cel al comenzilor adecvate, fie la acel al stărilor prevăzute. În această situaţie, apare necesitatea buclei de reacţie principiu prin care corpul conducător şi corpul condus se adaptează pentru a îndeplini comanda.

Trebuie observat că bucla de reacţie, cu toate că adaptează sistemul condus la comandă (în sensul eliminării unor perturbaţii), uneori reprezintă ea însăşi un canal prin care alte perturbaţii pătrund în sistem.

Rezumînd, putem spune că un sistem de comandă automat are două sarcini de îndeplinit: prima este aceea de a urmări obiectul condus, iar a doua este aceea de a determina comanda convenabilă atingerii scopului propus.

În cele ce urmează, vom prelua din automatică principiul corpului de comandă şi al corpului comandat, vom „contopi" cele două corpuri într-un singur tot pentru a realiza astfel o unitate elementară automată pe care o vom numi „bucla de reacţie de ordinul I". Din bucle de reacţie vom alcătui un ansamblu automat, care va putea reproduce sistemul real.

Unica condiţie pe care o vom pune în activitatea de modelare este aceea ca sistemul real, ales pentru a fi modelat, să poată fi astfel delimitat, în cadrul sistemelor de care este legat, încît să poată fi privit ca un sistem închis.

Pe modelul ce reprezintă sistemul real, vom căuta decizia optimă, adică prin simulări, sau prin aplicarea unui procedeu particular de calcul vom căuta să aflăm, la fiecare moment de timp, acele comenzi care fac maximul sau minimul unui criteriu de optim ales. Cînd vom realiza aceasta, vom spune că am găsit decizia optimă de conducere a sistemului real în timp.

Pentru a aduce la îndeplinire cele de mai sus vom avea de parcurs următoarele etape:

1° mai întîi va trebui să realizăm modelul matematic al sistemului real;

Scanned with OKEN Scanner

2° apoi va trebui să efectuăm o serie de simulări cu modelul pentru a studia evoluţia posibilă a diferitelor procese descrise de model;

3° în fine, va trebui să formulăm un criteriu de optim, care să reflecte cerinţele impuse sistemului, şi pe care îl vom „extremiza" printr-un algoritm adecvat.

În acest program de lucru, calculatorul automat va fi de o utilitate esenţială.

Pentru a realiza modelul sistemului vom folosi metoda de modelare dinamică datorată lui G. W. Forrester.

8.4. MODELAREA DINAMICĂ PRIN METODA LUI FORRESTER

Modelul matematic obţinut prin această metodă se prezintă sub forma unui sistem de ecuaţii cu diferenţe finite, formă sub care putem studia comportarea sistemului în timp.

Modul de scriere a ecuaţiilor este specific metodei; astfel, realizarea unui model se desfăşoară în trei faze:

1° întocmirea modelului sub forma de schemă grafică;

2° scrierea ecuaţiilor cu diferenţe finite şi

3° rezolvarea ecuaţiilor cu ajutorul unui sistem de calcul automat.

Pentru această metodă s-a creat (de către autorul ei: Forrester) un limbaj specializat, denumit *Dynamo*.

În figura 8.6 sînt prezentate simbolurile grafice ale metodei, împreună cu semnificaţia lor.

Să privim din nou figura 8.5, cu schema unui sistem automat, şi să-i revedem semnificaţia; o vom redesena, utilizînd simbolurile din figura 8.6. Vom obţine schema echivalentă din figura 8.7. Aici, corpul condus S are aspectul unui „rezervor" în care se „adună", sau se integrează, pînă la un anumit nivel, un *flux* provenit dintr-o sursă de natură cunoscută: oameni, materiale, informaţie etc. Corpul de comandă R, care este desenat ca un fel de robinet, reglează curgerea în rezervorul S. El îşi îndeplineşte funcţia de regulator pe baza a trei informaţii:

— nivelul prevăzut în S, fie el $x^*(t)$;

— nivelul curent în S, fie el $x(t)$, şi care îi parvine de la S prin bucla de reacţie;

— un parametru endogen al procesului de reglare, fie el TC.

Vom formula, acum, analitic procesul de reglare. Să presupunem că la sistemul real putem să anulăm treptat într-un mod liniar diferenţa care există între $x(t)$ şi $x^*(t)$. Dacă vom scrie că la fiecare moment de timp corecţia, care anulează diferenţa între mărimile de nivel curent şi prevăzut, este $x^*(t) - x(t)$ şi dacă punem şi condiţia de liniaritate, atunci obţinem pentru procesul de reglare formularea:

$$R(t) = k[x^*(t) - x(t)], \tag{8—29}$$

unde k este panta unei drepte şi înseamnă „cît de repede se anulează diferenţa". Evident acesta este un parametru legat de natura fizică a fluxului şi are dimensiunea $[T^{-1}]$. Astfel, dacă considerăm că $x(t)$ şi $x^*(t)$ reprezintă

Scanned with OKEN Scanner

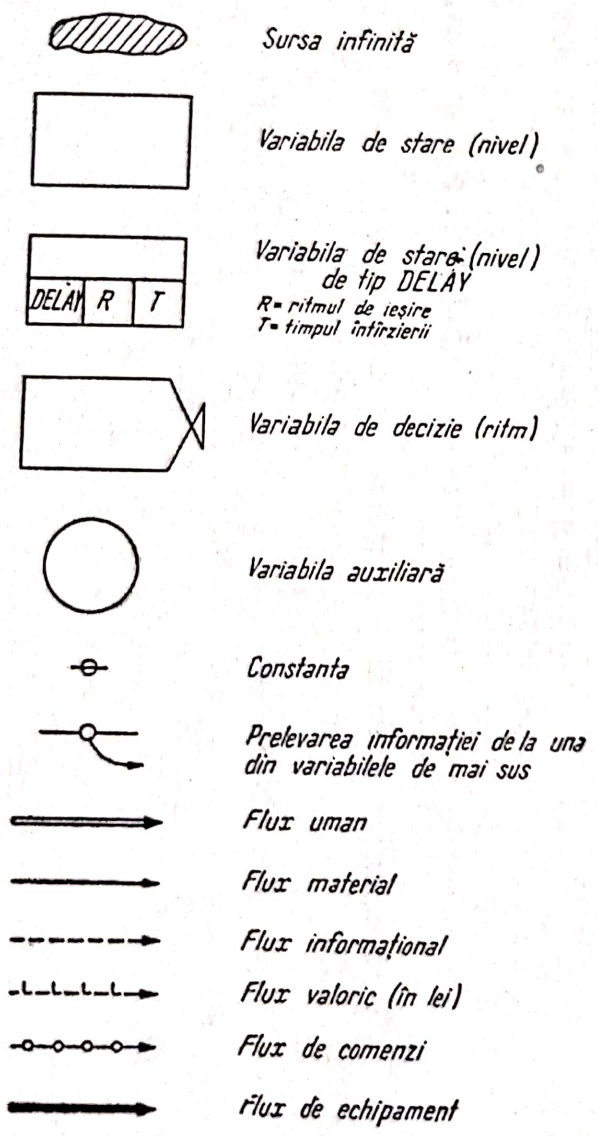


Fig. 8.6

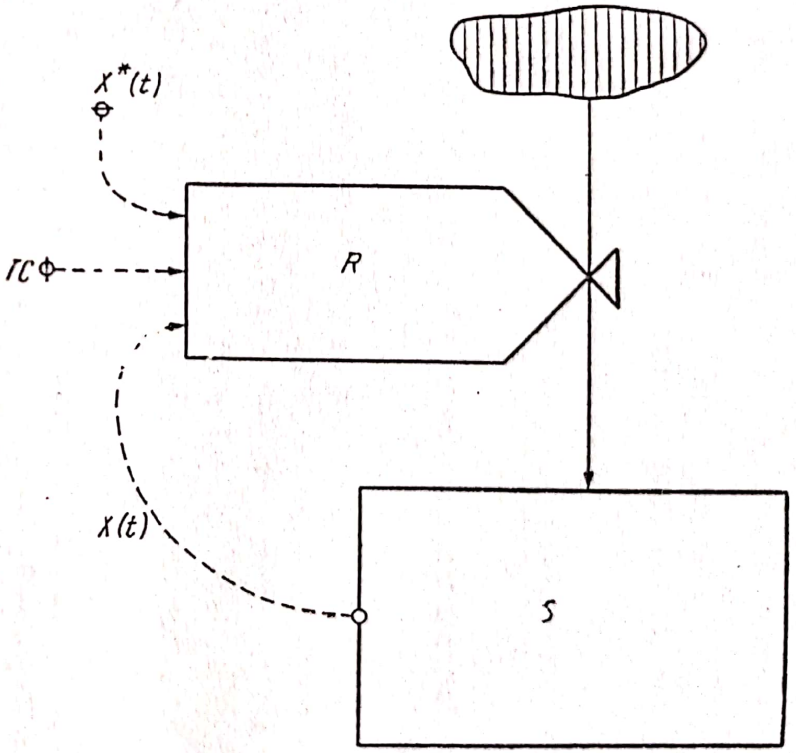


Fig. 8.7

Scanned with OKEN Scanner

dimensional o masă oarecare, de exemplu materii prime, atunci împărțind la timpul de corecție TC obținem numărul de unități materiale în unitatea de timp cu care se corectează diferența $x(t) - x^*(t)$.

Pentru acest exemplu, ecuația de reglare a fluxului este:

$$R(t) = \frac{1}{TC}\,[x^*(t) - x(t)] \qquad (8-30)$$

și se numește *ecuația de ritm* sau *de decizie* a corpului R. Observăm că în ecuația de ritm sînt prezente toate caracteristicile buclei de comandă automată:

— se reflectă starea sistemului: $x(t)$;

— este precizat scopul comenzii: $x^*(t)$;

— există o permanentă informare asupra stării corpului S datorită prezenței timpului t în ecuația (8—30). De fapt această prezență a lui t creează bucla de reacție.

Vom vedea în cele ce urmează și alte relații pentru R, liniare sau neliniare, precum și alte moduri în care putem formula decizia.

Datorită faptului că am conceput corpul S ca pe un rezervor, ecuația sa este prin definiție o ecuație de integrare a fluxului provenit de la corpul R, o ecuație care evidențiază, la fiecare moment de timp, valorile nivelului în S.

Ecuația corpului S este o ecuație cu diferențe finite și poartă numele de *ecuație de nivel*. Pentru exemplul ales vom putea scrie:

$$x(t + \Delta t) = x(t) + \Delta t \cdot R(t), \qquad (8-31)$$

unde: $R(t)$ este ritmul calculat cu ecuația (8—30), Δt este intervalul de calcul ales pentru tot modelul după teorema eșantionării. Practic, față de cea mai mică mărime de timp a modelului: $\frac{TC}{200} \leqslant \Delta t \leqslant \frac{TC}{2}$.

Într-un caz general ecuația de nivel se scrie:

$$x(t + \Delta t) = x(t) + \Delta t \cdot [R_1(t) - R_2(t)], \qquad (8-32)$$

unde $R_1(t)$ este ritmul de intrare în „rezervorul" S, iar $R_1(t)$ este ritmul de ieșire din S.

Trebuie observat, la descrierea modelului, că mărimile care intră într-un nivel să fie de aceeași natură cu mărimile care ies din același nivel. De asemenea, să fie respectată succesiunea ritm, nivel, ritm etc.

Adăugînd, relațiilor de mai sus, condiția inițială $x(0) = C$, sîntem în măsură să calculăm valorile lui $x(t)$ pentru o stare prevăzută $x^*(t)$ și un timp de corecție TC fixate.

În (8—31) avem o ecuație cu diferențe finite de ordinul I, care se poate integra prin metoda liniei poligonale a lui Euler. Integrarea ne va da la fiecare moment de timp, evoluția stării $x(t)$ și a ritmului $R(t)$.

Relațiile:

$$\begin{cases} x(0) = C \\ R(t) = \frac{1}{TC}\,[x^*(t) - x(t)] \\ x(t + \Delta t) = x(t) + \Delta t \cdot R(t) \end{cases} \qquad (8-33)$$

definesc **bucla negativă de reacție de ordinul întîi.**

Scanned with OKEN Scanner

Bucla se numește negativă deoarece eroarea este transformată în corecție și astfel intervine cu semn schimbat în ecuația de ritm.

Analiza ecuațiilor (8—33) ne arată că valoarea $R(t)$ a ritmului este de fapt o valoare medie pe durata unui interval de timp Δt, adică între două momente de calcul t și $t + \Delta t$.

Vom spune că ritmul există numai pe durata unui interval de timp Δt și-l vom nota simbolic R_{JK} sau R_{KL}, indicii JK, KL semnificînd intervalele de timp pe care s-a calculat ritmul. Nivelul este materializat, la fiecare moment de timp, de valorile variabilelor de stare ; vom pune în evidență felul diferit în care se calculează ritmul și nivelul prin notațiile indicilor :

$$X \cdot K = X \cdot J + (\Delta T)\, R \cdot JK \tag{8—34}$$

Această notație este cerută de compilatorul Dynamo. Ea pune în evidență calculul nivelului din moment în moment de timp și calculul ritmului din interval în interval de timp.

Ritmul are semnificația variației nivelului într-un interval de timp mic ; nivelul este cel care creează continuitatea sistemului între momentele de timp inițial și final ; integrează, deci, rezultatul ritmului.

Astfel trecînd la o reprezentare continuă a relațiilor (8—33) putem scrie :

$$x(t) = x(0) + \int_0^T R(t)\, dt, \tag{8—35}$$

cu soluția pentru un x^* fixat :

$$x(t) = x^* - (x^* - x_0)e^{-\frac{t}{T}}. \tag{8—36}$$

Totuși, această formă de exprimare nu ne este de mare utilitate la calculul tuturor variabilelor care intervin în model.

Ex. 8—4. Vom da, în continuare (lista 8.12), un program FORTRAN în care s-a transcris metoda de calcul a buclei negative de reacție de ordinul I. După cum se vede din lista 8.12, programul conține și două subrutine de trasare a graficelor la scări diferite.

În exemplul ales, am presupus că avem o rezervă de materie primă în exploatare (de exemplu un minereu oarecare) ce se acumulează către un nivel automatizat.

Rezultatele obținute sînt redate în lista 8.13 (tabelat) și în lista 8.14 (sub formă de grafice).

Acest model se poate aplica și evoluției numărului personalului dintr-un institut, creșterii biologice etc.

Bucla de reacție pozitivă reprezintă al doilea element de bază al modelării. Ea este asemănătoare, ca desen, cu bucla de reacție negativă (fig. 8.8), dar natura procesului de decizie este cu totul alta.

Ecuațiile buclei de reacție pozitivă sînt :

$$\begin{cases} x(0) = C \\ R(t) = x(t)/TC \\ x(t + \Delta t) = x(t) + \Delta t \cdot R(t) \end{cases} \tag{8—37}$$

Scanned with OKEN Scanner

```
C     LISTA 8.12
C     METODA DE CALCUL PENTRU SISTEMELE CU
C     BUCLA NEGATIVA DE REACTIE DE ORDINUL I  (EX. 8-4)
C
C     NDIM      DIMENSIUNEA VECTORILOR
C     TI        TIMPUL INITIAL
C     TF        TIMPUL FINAL
C     DT        INTERVALUL DE TIMP ALES PENTRU CALCUL
C     T1        TIMPUL DE CORECTIE AL ERORII IN RITMUL RN
C     CDT       COMANDA :STAREA PREVAZUTA
C     BRN       BUCLA DE REACTIE NEGATIVA
C     RN        RITMUL BUCLEI DE REACTIE NEGATIVE
      DOUBLE PRECISION ANTET
      DIMENSION ANTET(3),SCARA(3),CDA(51),BRN(51),RN(51),CARACTER(3)
      DATA NDIM/51/
      DATA C/7000/ BRN/1000/ TI/0 /TF/100/ DT/2/ T1/20/
      DATA CARACTER/'*    N    R'/
      DATA ANTET /' NIVEL * NIVEL N  RITM R'/SCARA/0,0,1/
      T=TI
      K=1
      GO TO 2
    1 CONTINUE
C     ECUATII DE NIVEL
C     MINEREU PRIMIT  N
      BRN(K)=BRN(J)+DT*RN(J)
    2 CONTINUE
C     ECUATII AUXILIARE
C     MINEREU COMANDAT  *
      CDA(K)=C
C     ECUATII DE RITM
C     RITMUL PRIMIRII  *
      RN(K)=(CDA(K)-BRN(K))/T1
      T=T+DT
      J=K
      K=K+1
      IF(K.LE.NDIM) GO TO 1
      CALL EDIT (CDA,NDIM,3,SCARA,TI,DT,ANTET,1,CARACTER,1)
      STOP
      END

      SUBROUTINE EDIT   (OO,N,NC,S,TI,PAS,GR,M1,M2,M3)
C
C     SUBRUTINA DE TRASARE DE GRAFICE LA SCARI DIFERITE
C        OO ADRESA VALORILOR - MATRICE DE DIMENSIUNE N * NC
C        N  NUMARUL DE VALORI ALE UNUI GRAFIC
C        NC NUMARUL DE GRAFICE
C        S  SCARA GRAFICELOR :
C              SE SPECIFICA O SCARA DIFERITA PRINTR-O VALOARE DIFERITA
C        TI TIMPUL INITIAL
C        PAS PASUL DE IMPRIMARE A TIMPULUI
C        GR NUMELE GRAFICELOR  VECTOR IN DUBLA PRECIZIE
C        M1 :  0   NU SE IMPRIMA CAROIAJUL
C           :  DIF SE IMPRIMA CAROIAJUL
C        M2 :  0   SE TRASEAZA CU SEMNELE STANDARD : CIFRE
C              DIF SE TRASEAZA CU SEMNELE DATE IN M2
C        M3 :  0   NU SE EDITEAZA VALORILE GRAFICELOR
C              DIF SE EDITEAZA VALORILE SUB FORMA DE TABELE
C        IA    PASUL CAROIAJULUI
C        NL    NUMARUL DE LINII IN PAGINA
C        ISC   PASUL SCARII
C
      LOGICAL L(10)
      REAL*8 GR(NC)
      DIMENSION OO(N,NC),S(NC),M2(NC),Y1(10),Y2(10),Y3(10),IC(10),IN(10)
      DIMENSION LIN(101),IAC(10),X(20),SAX(11),A(6),B(4)
      DATA IN/'1    2    3    4    5    6    7    8    9    0'/
      DATA IAY/'-'/IAO/'+'/IAX/'|'/IGOL/' '/
      DATA A/22H('1      T   '.   (A8.4X))/ B/1AH(F8.2,     G12.5)/
```

```
      II=10
      ISC=20
      IF(NC.GT.10) NC=10
      DO 1 I=1,NC
    1 L(I)=.FALSE.
      DO 2 I=1,101
    2 LIN(I)=IAX
      DO 3 I=1,101,ISC
    3 LIN(I)=IAO
      DO 4 I=1,NC
    4 IC(I)=IN(I)
      ASSIGN 28 TO LL
      IF (M1.EQ.0) ASSIGN 31 TO LL
      IF (M2(1).EQ.0) GO TO 6
      DO 5 I=1,NC
    5 IC(I)=M2(I)
    6 CONTINUE
      IF(M3.EQ.0) GO TO 8
C        EDITARE VALORI
      ENCODE( 2,7,A(4)) NC
      ENCODE( 2,7,B(3)) NC
    7 FORMAT(I2)
      T=TI
      NL=0
      DO 34  K=1,N
      IF(NL) 33,32,33
   32 NL=51
      WRITE(108,A) (GR(I),I=1,NC)
   33 NL=NL-1
      WRITE (108,B) T, (OO(K,I),I=1,NC)
      T=T+PAS
   34 CONTINUE
    8 CONTINUE
      PRINT 35
   35 FORMAT(/)
      DO 9 I=1,NC
      CALL MAXMIN(OO(1,I),N,X(2*I-1),X(2*I),IAC(1),IAC(2))
      T1=(IAC(1)-1)*PAS
      T2=(IAC(2)-1)*PAS
C        EDITARE EXTREME
    9 PRINT 10, IC(I),GR(I),X(2*I-1),T1,X(2*I),T2
   10 FORMAT      (' GRAFIC:',A1,A9,'   VALOAREA MAXIMA',G15.5,' PENTRU
     &T',G11.4,10X,'VALOAREA MINIMA',G15.5,' PENTRU T',G11.4)
      PRINT 23
   23 FORMAT('1',10X,'SCARA')
C        ANALIZA SCARA
      K=1
      I=1
      IF (NC.EQ.1) GO TO 12
      DO 11 J=2,NC
      IF(S(J).NE.S(J-1)) GO TO 12
      K=K+1
   11 CONTINUE
   12 CALL MAXMIN(X(2*I-1),2*K,YMAX,YMIN,IMAX,IMIN)
   13 Y1(I)=(YMAX-YMIN)/100
      Y2(I)=YMAX
      Y3(I)=YMIN
      IF(I.GE.NC) GO TO 14
      I=I+1
      IF(K.EQ.1) GO TO 11
      K=K-1
      GO TO 13
   14 CONTINUE
      DO 21 I=1,NC
      IF(Y2(I).NE.Y3(I)) GO TO 18
      L(I)=.TRUE.
      DO 15 J=1,6
   15 SAX(J)=IGOL
      IF(Y2(I)-W) 16,16,17
```

Scanned with OKEN Scanner

```
   16 JJ=1
      SAX(1)=Y3(I)
      GO TO 21
   17 JJ=41
      SAX(3)=Y3(I)
      GO TO 21
   18 SAX(1)=Y3(I)
      DO 19 JV=2,6
   19 SAX(JV)=SAX(JV-1)+Y1(I)*ISC
   20 FORMAT(13X,A1,7X,G11.5,5G20.5)
C        EDITARE SCARA
   21 PRINT 20,IC(I),(SAX(J),J=1,6)
C        EDITARE GRAFICE
      T=TI
      DO 31 I=1,N
      DO 22 J=1,10
   22 IAC(J)=IGOL
      DO 26 K=1,NC
      IF(L(K)) GO TO 24
      JJ=ABS(OO(I,K)-Y3(K))/Y1(K)+1.4
   24 IF(LIN(JJ).EQ.IGOL.OR.LIN(JJ).EQ.IAX.OR.LIN(JJ).EQ.IA0) GO TO 25
      IAC(K)=IC(K)
      GO TO 26
   25 LIN(JJ)=IC(K)
   26 CONTINUE
      PRINT 30,T,IAC,LIN
      T=T+PAS
      DO 27 K=1,101
   27 LIN(K)=IGOL
      GO TO LL
   28 IF(I.NE.II) GO TO 31
      II=II+10
      DO 29 JP=1,101,ISC
   29 LIN(JP)=IA0
   30 FORMAT( F8.2,'I',10A1,'I',2X,101A1)
   31 CONTINUE
      RETURN
      END

* FETCHS LN:SUBMOD,DV:RD1,GN:1,VN:1,FN:MAXMIN
      SUBROUTINE MAXMIN(TF,N,P,R,IP,IR)
C
C                 SUBRUTINA DE AFLARE A MAX SI MIN UNUI VECTOR
C
C       TF        VECTOR DE CERCETAT
C       N         NUMARUL DE VALORI
C       P       * MAXIM
C       R       * MINIM
C       IP      * PUNCTUL DE MAXIM
C       IR      * PUNCTUL DE MINIM
C
C
      DIMENSION TF(N)
      P=TF(1)
      R=TF(1)
      IP=1
      IR=1
      IF(N.LE.1) RETURN
      DO 5 I=2,N
      IF(P-TF(I))2,3,3
    2 P=TF(I)
      IP=I
    3 IF(R-TF(I))5,5,4
    4 R=TF(I)
      IR=I
    5 CONTINUE
      RETURN
      END
```

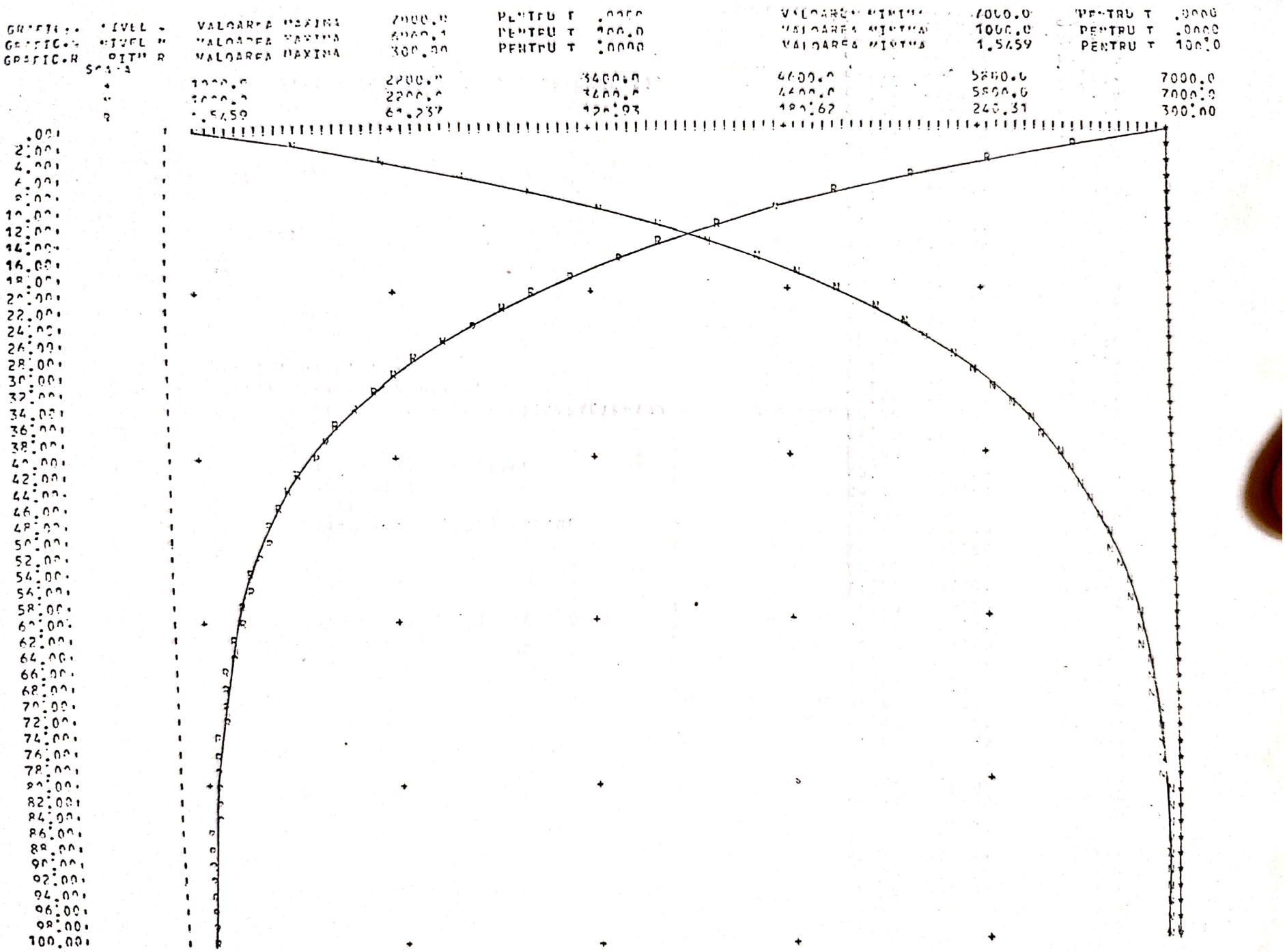
Lista 8.12

Scanned with OKEN Scanner

T	NIVEL *	NIVEL N	RITM R
0.00	7000.0	1000.0	300.00
2.00	7000.0	1600.0	270.00
4.00	7000.0	2140.0	243.00
6.00	7000.0	2626.0	218.70
8.00	7000.0	3063.4	196.83
10.00	7000.0	3457.1	177.15
12.00	7000.0	3811.4	159.43
14.00	7000.0	4130.2	143.49
16.00	7000.0	4417.2	129.14
18.00	7000.0	4675.5	116.23
20.00	7000.0	4907.9	104.60
22.00	7000.0	5117.1	94.143
24.00	7000.0	5305.4	84.729
26.00	7000.0	5474.9	76.256
28.00	7000.0	5627.4	68.630
30.00	7000.0	5764.7	61.767
32.00	7000.0	5888.2	55.591
34.00	7000.0	5999.4	50.032
36.00	7000.0	6099.4	45.029
38.00	7000.0	6189.5	40.526
40.00	7000.0	6270.5	36.473
42.00	7000.0	6343.5	32.826
44.00	7000.0	6409.1	29.543
46.00	7000.0	6468.2	26.589
48.00	7000.0	6521.4	23.930
50.00	7000.0	6569.3	21.537
52.00	7000.0	6612.3	19.383
54.00	7000.0	6651.1	17.445
56.00	7000.0	6686.0	15.700
58.00	7000.0	6717.4	14.130
60.00	7000.0	6745.7	12.717
62.00	7000.0	6771.1	11.446
64.00	7000.0	6794.0	10.301
66.00	7000.0	6814.6	9.2709
68.00	7000.0	6833.1	8.3437
70.00	7000.0	6849.8	7.5094
72.00	7000.0	6864.8	6.7584
74.00	7000.0	6878.3	6.0826
76.00	7000.0	6890.5	5.4744
78.00	7000.0	6901.5	4.9270
80.00	7000.0	6911.3	4.4343
82.00	7000.0	6920.2	3.9908
84.00	7000.0	6928.2	3.5918
86.00	7000.0	6935.3	3.2326
88.00	7000.0	6941.8	2.9094
90.00	7000.0	6947.6	2.6184
92.00	7000.0	6952.9	2.3564
94.00	7000.0	6957.6	2.1207
96.00	7000.0	6961.8	1.9086
98.00	7000.0	6965.6	1.7178
100.00	7000.0	6969.1	1.5459

Lista 8.13

Scanned with OKEN Scanner



Lista 8.14

```
C       LISTA 8.15
C       METODA DE CALCUL PENTRU SISTEMELE CU
C       BUCLA POZITIVA DE REACTIE DE ORDINUL I   (EX. 8-5)
C
C       NDIM      DIMENSIUNEA VECTORILOR
C       TI        TIMPUL INITIAL
C       TF        TIMPUL FINAL
C       DT        INTERVALUL DE TIMP ALES PENTRU CALCUL
C       T2        TIMPUL RITMULUI RP
C       BRP       BUCLA DE REACTIE POZITIVA
C       RP        RITMUL BUCLEI DE REACTIE POZITIVE
        DOUBLE PRECISION ANTET
        DATA NDIM/51/
        DIMENSION ANTET(2),SCARA(2),BRP(51),RP(51),CARACTER(2)
        DATA BRP/20/ TI/0/TF/100/ DT/2/ T2/20/
        DATA ANTET /' PERS.A.  RITM.A.'/
        DATA CARACTER/'P    R'/
        K=1
        GO TO 2
      1 CONTINUE
C       ECUATII DE NIVEL
C       PERSONAL ANGAJAT
        BRP(K)=BRP(J)+DT*RP(J)
      2 CONTINUE
C       ECUATII DE RITM
C       RITMUL    ANGAJARII
        RP(K)=BRP(K)/T2
        J=K
        K=K+1
        IF(K.LE.NDIM) GO TO 1
        CALL EDIT (BRP,NDIM,2,SCARA,TI,DT,ANTET,1,CARACTER,1)
        STOP
        END
```

Lista 8.15

T	PERS.A.	RITM.A.
.00	20.000	-.94540E-29
2.00	22.000	-.36030E-31
4.00	24.200	1.0000
6.00	26.620	1.1000
8.00	29.282	1.2100
10.00	32.210	1.3310
12.00	35.431	1.4641
14.00	38.974	1.6105
16.00	42.872	1.7716
18.00	47.159	1.9487
20.00	51.875	2.1436
22.00	57.062	2.3579
24.00	62.769	2.5937
26.00	69.045	2.8531
28.00	75.950	3.1384
30.00	83.545	3.4523
32.00	91.899	3.7975
34.00	101.09	4.1772
36.00	111.20	4.5950
38.00	122.32	5.0545
40.00	134.55	5.5599
42.00	148.01	6.1159
44.00	162.81	6.7275
46.00	179.09	7.4003
48.00	196.99	8.1403
50.00	216.69	8.9543
52.00	238.36	9.8497
54.00	262.20	10.835
56.00	288.42	11.918
58.00	317.26	13.110
60.00	348.99	14.421
62.00	383.89	15.863
64.00	422.28	17.449
66.00	464.50	19.194
68.00	510.95	21.114
70.00	562.05	23.225
72.00	618.25	25.548
74.00	680.08	28.102
76.00	748.09	30.913
78.00	822.90	34.004
80.00	905.19	37.404
82.00	995.70	41.145
84.00	1095.3	45.259
86.00	1204.8	49.785
88.00	1325.3	54.764
90.00	1457.8	60.240
92.00	1603.6	66.264
94.00	1764.0	72.891
96.00	1940.3	80.180
98.00	2134.4	88.198
100.00	2347.8	97.017

Lista 8.16

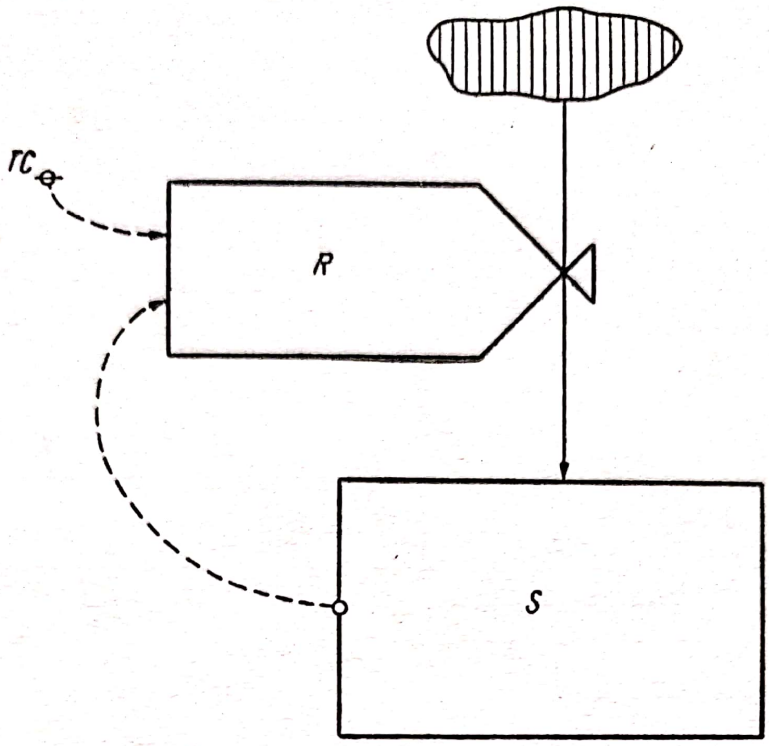


Fig. 8.8

cu aceeaşi semnificaţie pentru mărimile ce intervin aici ca şi la bucla negativă. În ecuaţia de ritm se observă absenţa mărimii de stare prevăzută $x^*(t)$. Acest lucru schimbă natura deciziei, în sensul că valoarea curentă a stării $x(t)$ nu se mai îndreaptă către starea prevăzută ; nu mai există, deci, o corecţie a stării, ci starea creşte în mod exponenţial faţă de valorile anterioare.

Astfel de fenomene sînt observate la diviziunea celulară, exploziile atomice în lanţ, creşterea cunoştinţelor tehnice etc.

În natură buclele de reacţie pozitive sînt întîlnite în combinaţii cu buclele de reacţie negative care vor frîna explozia exponenţială a celor dintîi.

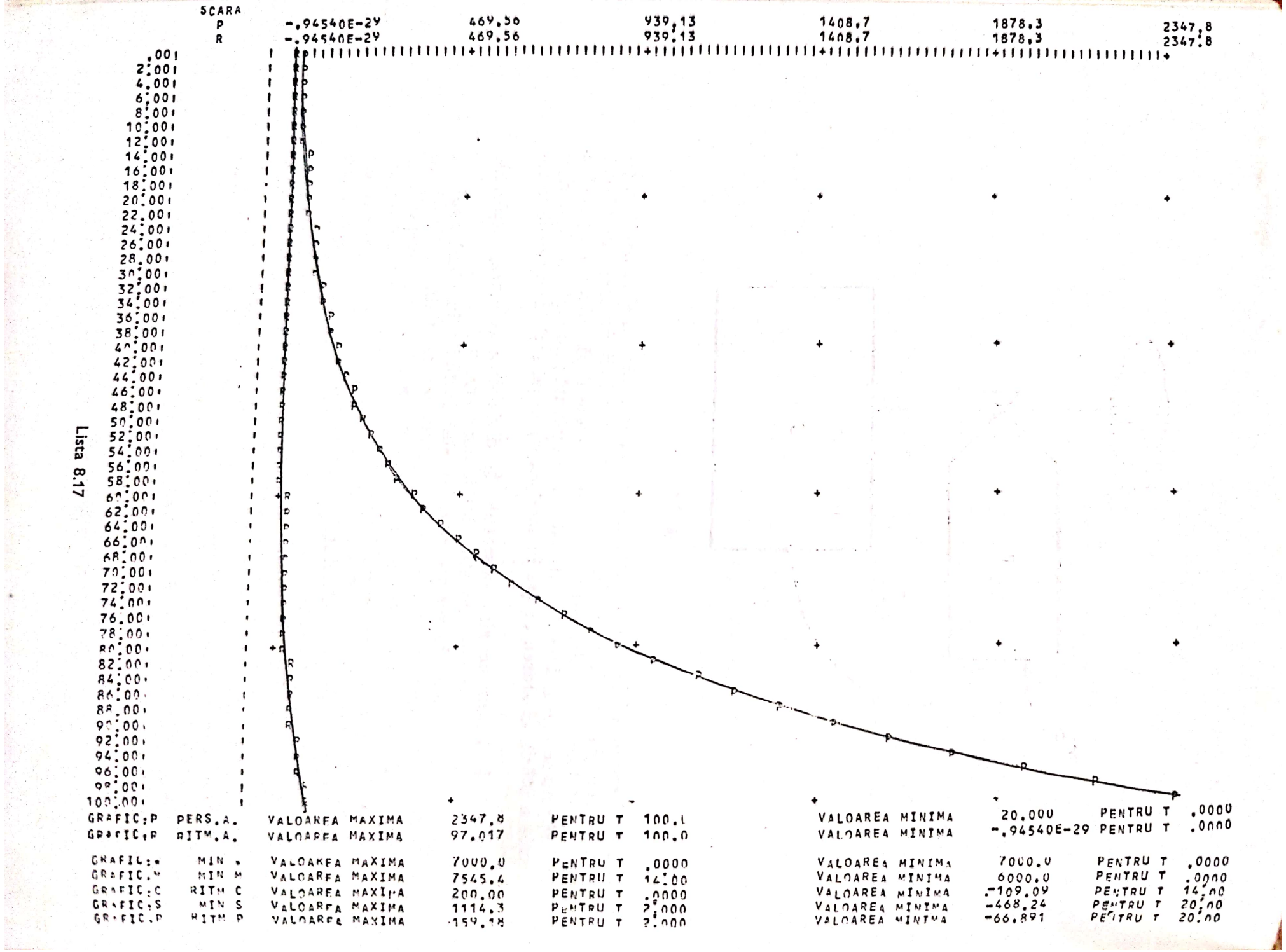
Ex. 8—5. Vom da un exemplu de calcul al unei bucle de reacţie pozitive. În acest exemplu, mărimea de stare $x(t)$ are semnificaţia unui număr de oameni care sînt atraşi într-o activitate nou creată. Se presupune că cei ocupaţi înainte în această activitate îi învaţă pe cei nou angajaţi, lucru care provoacă această dezvoltare exponenţială a numărului lor total.

Programul principal de calcul este redat în lista 8.15; subrutinele apelate sînt cele din lista 8.12, prezentată anterior. Rezultatele sînt indicate în lista 8.16 (tabelat) şi în lista 8.17 (sub formă de grafic).

Bucla de reacţie negativă de ordinul II este o combinaţie între două bucle de reacţie de ordinul I, una negativă şi alta pozitivă, rolul dominant revenind buclei de reacţie negative (fig. 8.9).

Analitic ea este reprezentată de un sistem de două ecuaţii cu diferenţe finite.

Bucla de reacţie pozitivă are rolul de a introduce o întîrziere între sistemul de conducere şi sistemul condus al buclei de reacţie negativă,



Lista 8.17

GRAFIC:P	PERS.A.	VALOAREA MAXIMA	2347.8	PENTRU T	100.0	VALOAREA MINIMA	20.000	PENTRU T .0000
GRAFIC:P	RITM.A.	VALOAREA MAXIMA	97.017	PENTRU T	100.0	VALOAREA MINIMA	-.94540E-29	PENTRU T .0000
GRAFIC:*	MIN *	VALOAREA MAXIMA	7000.0	PENTRU T	.0000	VALOAREA MINIMA	7000.0	PENTRU T .0000
GRAFIC:M	MIN M	VALOAREA MAXIMA	7545.4	PENTRU T	14.00	VALOAREA MINIMA	6000.0	PENTRU T .0000
GRAFIC:C	RITM C	VALOAREA MAXIMA	200.00	PENTRU T	.0000	VALOAREA MINIMA	-109.09	PENTRU T 14.00
GRAFIC:S	MIN S	VALOAREA MAXIMA	1114.3	PENTRU T	2.000	VALOAREA MINIMA	-468.24	PENTRU T 20.00
GRAFIC:P	RITM P	VALOAREA MAXIMA	159.18	PENTRU T	2.000	VALOAREA MINIMA	-66.891	PENTRU T 20.00

Scanned with OKEN Scanner

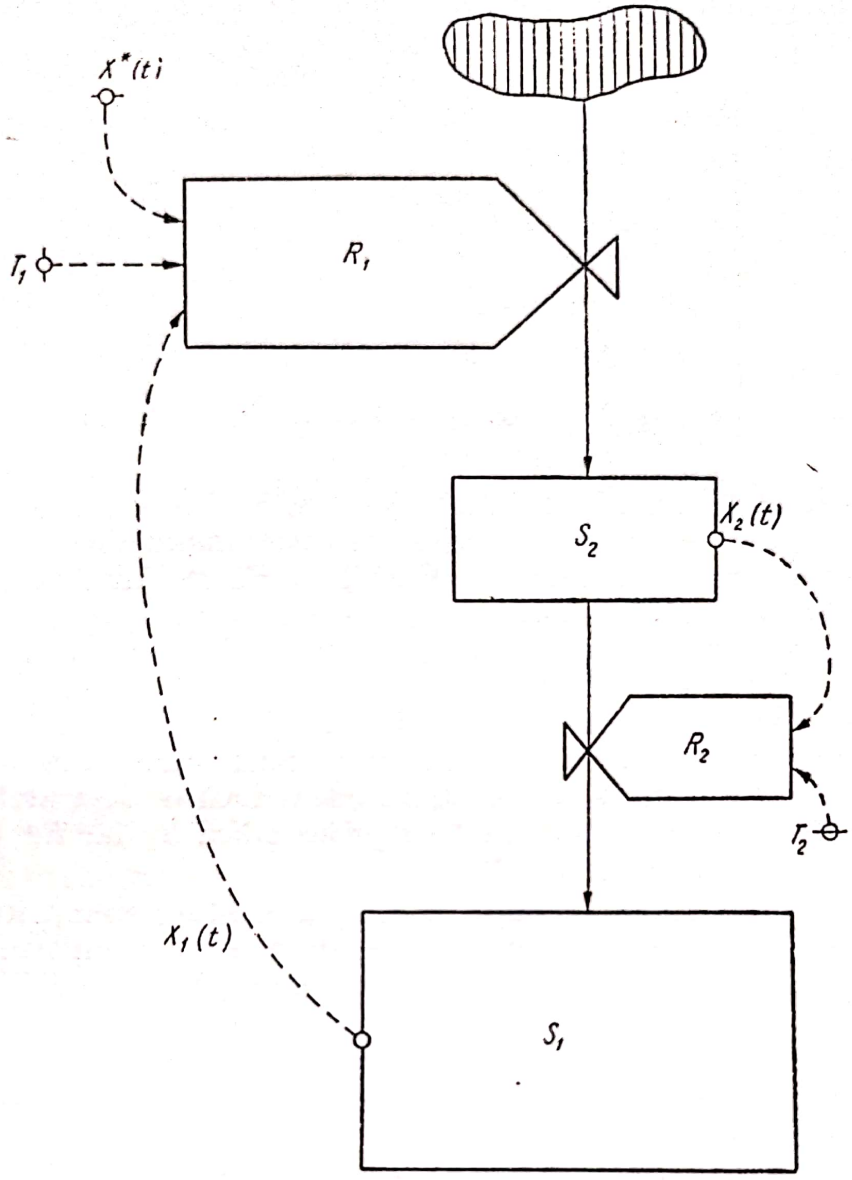


Fig. 8.9

întîrziere care dă valorilor nivelului $x_1(t)$ un aspect oscilant în jurul valorilor nivelului impus $x^*(t)$. Astfel, dacă reluînd modelul buclei de reacţie negativă, S_1 reprezintă un nivel de materii prime, iar R_1 este ritmul care comandă fluxul lor, putem presupune că primirea lor în nivel nu se face instantaneu, ci există o întîrziere între comanda $R_1(t)$ şi primirea efectivă a fluxului în nivelul $x_1(t)$. Atunci S_2 reprezintă un nivel ajutător, în care materiile prime se strîng înainte de a ajunge în S_1. S_2 este rezervorul materiilor prime comandate şi în curs de primire, iar R_2 este ritmul efectiv de primire.

Putem interpreta nivelul S_2 ca un nivel de comenzi cu rol intermediar, dar în acest caz va trebui să modificăm desenul, deoarece prin definiţie un nivel trebuie să integreze fluxuri de aceeaşi natură.

Alegerea timpilor T_1 şi T_2 este esenţială pentru apropierea modelului de o realitate fizică.

Scanned with OKEN Scanner

Dăm, în continuare, formularea ecuațiilor buclei de reacție negative de ordinul II:

$$\begin{cases} x_1(0) = C_1 \\ x_2(0) = C_2 \\ R_1(t) = \dfrac{x^*(t) - x_1(t)}{T_1} \\ R_2(t) = x_2(t)/T_2 \\ x_1(t + \Delta t) = x_1(t) + \Delta t \cdot R_2(t) \\ x_2(t + \Delta t) = x_2(t) + \Delta t \cdot [R_1(t) - R_2(t)] \end{cases} \tag{8—38}$$

Ex. 8—6. Programul FORTRAN (lista 8.18) și rezultatele (lista 8.19 — tabelat și lista 8.20 — grafic) se referă la un exemplu asemănător cu cele din *Ex. 8—4* și *Ex. 8—5*. Toate explicațiile necesare sînt date prin comentariile din lista 8.18. Mai precizăm, referindu-ne la notațiile (identificatorii) folosiți în programele precedente (v. lista 8.18), că: nivelul principal *BRN* (adică x_1) are semnificația cantității de materie primă (pentru cazul din *Ex. 8—6*: minereuri care sînt primite la uzină); nivelul secundar *BRP* (adică x_2) are semnificația minereurilor în primire, care au fost comandate conform cererii *CDT* (adică x^*) și nu au intrat încă în depozitele de destinație; *RN* este ritmul cererii de materie primă adică R_1, iar *RP* este ritmul primirii materiei prime, adică R_2.

Observăm că în cursul evoluției în timp a acestui sistem, ritmurile *RN* și *RP* iau alternativ valori pozitive și negative, acest comportament fiind dictat de necesitatea de a aduce valoarea nivelului principal *BRN* la valoarea comenzii *CDT*. Putem interpreta valorile negative ale ritmurilor ca o decizie de respingere a fluxului de materie primă în supraabundență cu presupunerea de depozitare a lor în alt loc decît cel rezervat, *BRP* (adică x_2) sau *BRN* (adică x_1).

Este caracteristic pentru bucla de reacție de ordinul II acest comportament oscilant, adică aducerea valorii variabilei de stare la valoarea comenzii prin corecții pozitive alternate cu corecții negative.

În felul acesta bucla negativă de reacție de ordinul II se deosebește de bucla negativă de reacție de ordinul I, la care corectarea valorilor variabilei de stare se face treptat către valoarea comenzii, întotdeauna în același sens.

Am văzut că în bucla de reacție negativă de ordinul II se introduce, printr-o buclă pozitivă cuprinsă în interiorul ei, o întîrziere între acțiunea ritmului buclei negative și cea a nivelului corespunzător, întîrziere care dă valorilor nivelului buclei un aspect caracteristic.

Cel care provoacă întîrzierea este ritmul R_2, numit și ritmul întîrziat al lui R_1. Pentru momentul K, ritmul corespunzător, adică $R \cdot KL$ (exprimat în termenii limbajului Dynamo), are întotdeauna forma:

$$R \cdot KL = \frac{NIV \cdot K}{TD}, \tag{8—39}$$

Scanned with OKEN Scanner

```
C     LISTA 8.18
C     METODA DE CALCUL PENTRU SISTEMELE CU
C     BUCLA NEGATIVA DE REACTIE DE ORDINUL II (EX. 3-6)
C     MODELUL DINAMIC AL CERERII-PRIMIRII MATERIILOR PRIME
C     IN CONDITIILE UNEI COMENZI CONSTANTE
C
C     NDIM      DIMENSIUNEA VECTORILOR
C     TI        TIMPUL INITIAL
C     TF        TIMPUL FINAL
C     DT        INTERVALUL DE TIMP ALES PENTRU CALCUL
C                     DT < 1/2*(MIN(T1,T2))
C     T1        TIMPUL DE CORECTIE AL ERORII IN RITMUL RN
C     T2        TIMPUL DE TRAVERSARE A INTIRZIERII
C     CDT       COMANDA :STAREA PREVAZUTA
C     BRN       BUCLA DE REACTIE NEGATIVA
C     RN        RITMUL BUCLEI DE REACTIE NEGATIVE
C     BRP       BUCLA DE REACTIE POZITIVA
C     RP        RITMUL BUCLEI DE REACTIE POZITIVE
      DOUBLE PRECISION ANTET
      DIMENSION ANTET(5),SCARA(5),CDA(51),BRN(51),RN(51),BRP(51),RP(51)
     *,CARACTER(5)
      DATA NDIM /51/CARACTER/'*   M   C   S   P'/SCARA /1.,1.,0.,0.,0./
      DATA C/7000/ BRN/6000/ BRP/1000/ TI/0/ TF/100/DT/2/ T1/5/ T2/7/
      DATA ANTET/'   MIN *   MIN M  RITM C   MIN S  RITM P'/
      T=TI
      K=1
      GO TO 2
    1 CONTINUE
C     ECUATII DE NIVEL
C     MINEREU PRIMIT  M
      BRN(K)=BRN(J)+DT*RP(J)
C     MINEREU IN SOSIRE S
      BRP(K)=BRP(J)+DT*(RN(J)-RP(J))
    2 CONTINUE
C     ECUATII AUXILIARE
C     MINEREU COMANDAT  *
      CDA(K)=C
C     ECUATII DE RITM
C     RITMUL CERERII  C
      RN(K)=(CDA(K)-BRN(K))/T1
C     RITMUL PRIMIRII  P
      RP(K)=BRP(K)/T2
      J=K
      K=K+1
      T=T+DT
      IF(K.LE.NDIM) GO TO 1
      CALL EDIT (CDA,NDIM,5,SCARA,TI,DT,ANTET,1,CARACTER,1)
      STOP
      END
```

Lista 8.18

Scanned with OKEN Scanner

T	MIN *	MIN M	RITM C	MIN S	RITM P
.00	7000.0	6000.0	200.00	1000.0	142.86
2.00	7000.0	6285.7	142.86	1114.3	159.18
4.00	7000.0	6604.1	79.184	1081.6	154.52
6.00	7000.0	6913.1	17.376	930.96	132.99
8.00	7000.0	7179.1	-35.822	699.72	99.961
10.00	7000.0	7379.0	-75.806	428.16	61.166
12.00	7000.0	7501.4	-100.27	154.22	22.031
14.00	7000.0	7545.4	-109.09	-90.391	-12.913
16.00	7000.0	7519.6	-103.92	-282.74	-40.391
18.00	7000.0	7438.8	-87.764	-409.79	-58.542
20.00	7000.0	7321.7	-64.347	-468.24	-66.891
22.00	7000.0	7188.0	-37.591	-463.15	-66.164
24.00	7000.0	7055.6	-11.125	-406.00	-58.000
26.00	7000.0	6939.6	12.075	-312.25	-44.607
28.00	7000.0	6850.4	29.918	-198.89	-28.412
30.00	7000.0	6793.6	41.283	-82.226	-11.747
32.00	7000.0	6770.1	45.981	23.833	3.4046
34.00	7000.0	6776.9	44.620	108.99	15.569
36.00	7000.0	6808.0	38.391	167.09	23.869
38.00	7000.0	6855.8	28.844	196.13	28.019
40.00	7000.0	6911.8	17.636	197.78	28.254
42.00	7000.0	6968.3	6.3344	176.54	25.221
44.00	7000.0	7018.8	-3.7539	138.77	19.824
46.00	7000.0	7058.4	-11.684	91.615	13.088
48.00	7000.0	7084.6	-16.919	42.072	6.0103
50.00	7000.0	7096.6	-19.323	-3.7862	-.54089
52.00	7000.0	7095.5	-19.106	-41.350	-5.9071
54.00	7000.0	7083.7	-16.744	-67.748	-9.6783
56.00	7000.0	7064.4	-12.873	-81.879	-11.697
58.00	7000.0	7041.0	-8.1938	-84.230	-12.033
60.00	7000.0	7016.9	-3.3805	-76.552	-10.936
62.00	7000.0	6995.0	.99375	-61.441	-8.7773
64.00	7000.0	6977.5	4.5047	-41.899	-5.9856
66.00	7000.0	6965.5	6.8992	-20.718	-2.9883
68.00	7000.0	6959.5	8.0945	-1.1433	-.16332
70.00	7000.0	6959.2	8.1602	15.372	2.1961
72.00	7000.0	6963.6	7.2820	27.501	3.9001
74.00	7000.0	6971.4	5.7219	34.065	4.8664
76.00	7000.0	6981.1	3.7750	35.776	5.1108
78.00	7000.0	6991.3	1.7305	33.104	4.7291
80.00	7000.0	7000.8	-.16094	27.107	3.8724
82.00	7000.0	7008.6	-1.7102	19.040	2.7200
84.00	7000.0	7014.0	-2.7984	10.180	1.4542
86.00	7000.0	7016.9	-3.3805	1.6743	.23919
88.00	7000.0	7017.4	-3.4758	-5.5950	-.79500
90.00	7000.0	7015.8	-3.1578	-10.927	-1.5609
92.00	7000.0	7012.7	-2.5336	-14.120	-2.0172
94.00	7000.0	7008.6	-1.7266	-15.153	-2.1647
96.00	7000.0	7004.3	-.86094	-14.277	-2.0395
98.00	7000.0	7000.2	-.45313E-01	-11.920	-1.7028
100.00	7000.0	6996.8	.63594	-8.6346	-1.2292

Lista 8.19

Scanned with OKEN Scanner

unde: $NIV \cdot K$ este nivelul la momentul K, iar TD reprezintă timpul mediu al întîrzierii.

Sensul ritmului întîrziat în modelarea dinamică este acela de decizie implicită sau de reacţie după stare.

În funcţie de natura procesului modelat, putem introduce două, trei sau mai multe nivele ajutătoare între corpul conducător şi corpul condus, pentru a obţine un aspect cît mai apropiat de realitate pentru ritmul de ieşire final.

Totuşi nu introducerea multor nivele schimbă aspectul ritmului final, ci constanta de timp a întîrzierii. Astfel, pentru uşurinţa folosirii în procesul modelării a acestui procedeu s-a realizat aşa numita „funcţie de întîrziere" care este o **întîrziere de ordinul III tipizată,** denumită (în limbaj Dynamo) **DELAY.** Prin alegerea adecvată a constantei de timp putem obţine efectul unei întîrzieri de ordinul n.

Dăm în figura 8.10 schema grafică a întîrzierii de ordinul III şi, mai jos, ecuaţiile ei în limbaj Dynamo (*DELAY3*):

DELAY3

— valori iniţiale:

$$L1 = (Q)\,(DEL/3)$$

$$L2 = (Q)\,(DEL/3)$$

$$L3 = (Q)\,(DEL/3)$$

unde DEL este constanta de timp a întîrzierii;

— ecuaţii de ritm:

$$R1.KL = L1.K/(DEL/3)$$

$$R2.KL = L2.K/(DEL/3)$$

$$R.KL = L3.K/(DEL/3)$$

— ecuaţii de nivel:

$$L1.K = L1.J + (DT)(Q.JK - R1.JK)$$

$$L2.K = L2.J + (DT)(R1.JK - R2.JK)$$

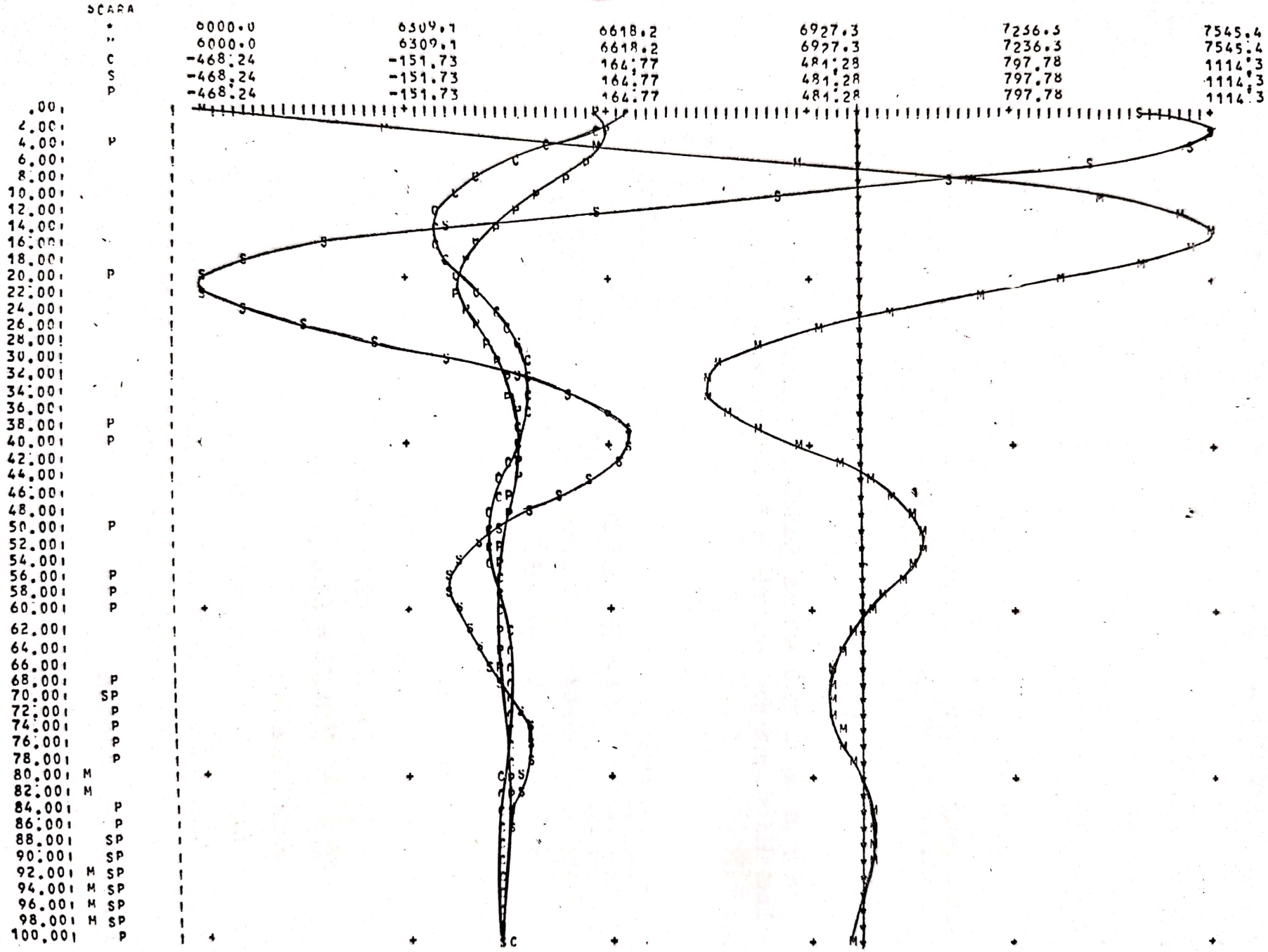
$$L3.K = L3.J + (DT)(R2.JK - R.JK)$$

ritmul de ieşire fiind R.

În lista 8.21 prezentăm o subrutină FORTRAN pentru calculul funcţiei *DELAY*.

```
GRAFIC:*   MIN *   VALOAREA MAXIMA   7000.0   PENTRU T  .0000   VALOAREA MINIMA   7000.0   PENTRU T  .0000
GRAFIC:M   MIN M   VALOAREA MAXIMA   7545.4   PENTRU T  14.00   VALOAREA MINIMA   6000.0   PENTRU T  .0000
GRAFIC:C   RITM C  VALOAREA MAXIMA   200.00   PENTRU T  .0000   VALOAREA MINIMA   -109.09  PENTRU T  14.00
GRAFIC:S   MIN S   VALOAREA MAXIMA   1114.3   PENTRU T  2.000   VALOAREA MINIMA   -468.24  PENTRU T  20.00
GRAFIC.P   RITM P  VALOAREA MAXIMA   159.18   PENTRU T  2.000   VALOAREA MINIMA   -66.891  PENTRU T  20.00
```

Lista 8.20



Lista 8.20 (continuare)

Scanned with OKEN Scanner

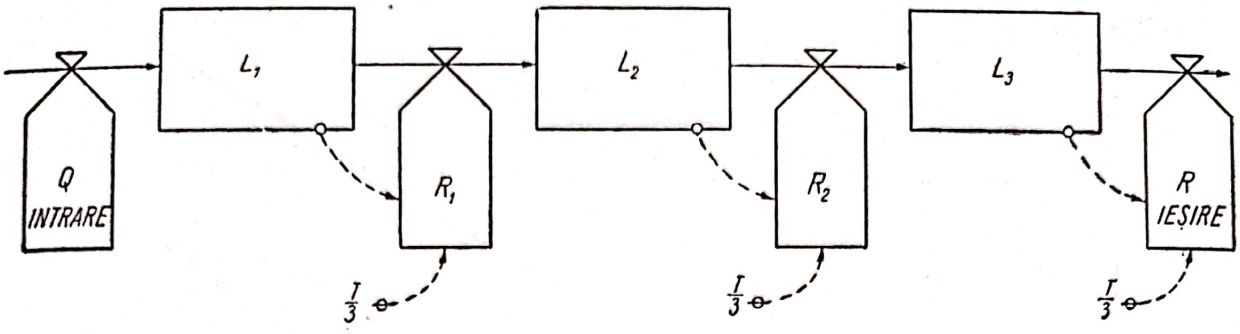


Fig. 8.10

Redăm, în continuare, diferite aspecte grafice ale acestui ritm de ieşire, pentru intrări „standard" (de tip „rampă", „treaptă" şi „sinusoidal"):

— intrare rampă în figura 8.11;

— intrare treaptă în figura 8.12. Aici punctul de inflexiune al curbei are abscisa $t = T - \frac{T}{n}$, unde n este ordinul întîrzierii. Pentru $n \to \infty$ se reproduce intrarea treaptă, dar după un interval de timp T;

— intrare sinusoidală în figura 8.13.

Altă funcţie întîlnită în procedeele de modelare este *nivelarea exponenţială medie a ritmurilor*. Ea se aplică ritmurilor oscilante pe care le nivelează transformîndu-le în nivele. Nivelul păstrează dimensiunile ritmului din care provine. Funcţia descrie de fapt o buclă de reacţie negativă de ordinul I. Dăm mai jos forma tipică a ecuaţiei de nivelare exponenţială (folosind, de asemenea, notaţiile de limbaj Dynamo):

$$RN.K = RN.J + DT(R.JK - RN.J)/TN$$

```
      FUNCTION DELAY(Q,L,DT,TIME)
C
C                  SUBRUTINA DE INTIRZIERE ORD III -FORRESTER
C        Q     RITMUL DE INTRARE
C        L     NIVELE AJUTATOARE (3) INITIALIZATE IN SUBRUTINA
C        DT    INTERVALUL DE CALCUL AL SISTEMULUI
C        TIME  TIMPUL INTIRZIERII
C
      REAL L(4)
      T=TIME/3
      IF (L(4).EQ.1) GO TO 2
      L(4)=1
      DO 1 I=1,3
    1 L(I)=Q*T
    2 CONTINUE
      R1=L(1)/T
      R2=L(2)/T
      R=L(3)/T
      L(1)=L(1)+DT*(Q-R1)
      L(2)=L(2)+DT*(R1-R2)
      L(3)=L(3)+DT*(R2-R)
      DELAY=R
      RETURN
      END
```

Lista 8.21

Scanned with OKEN Scanner

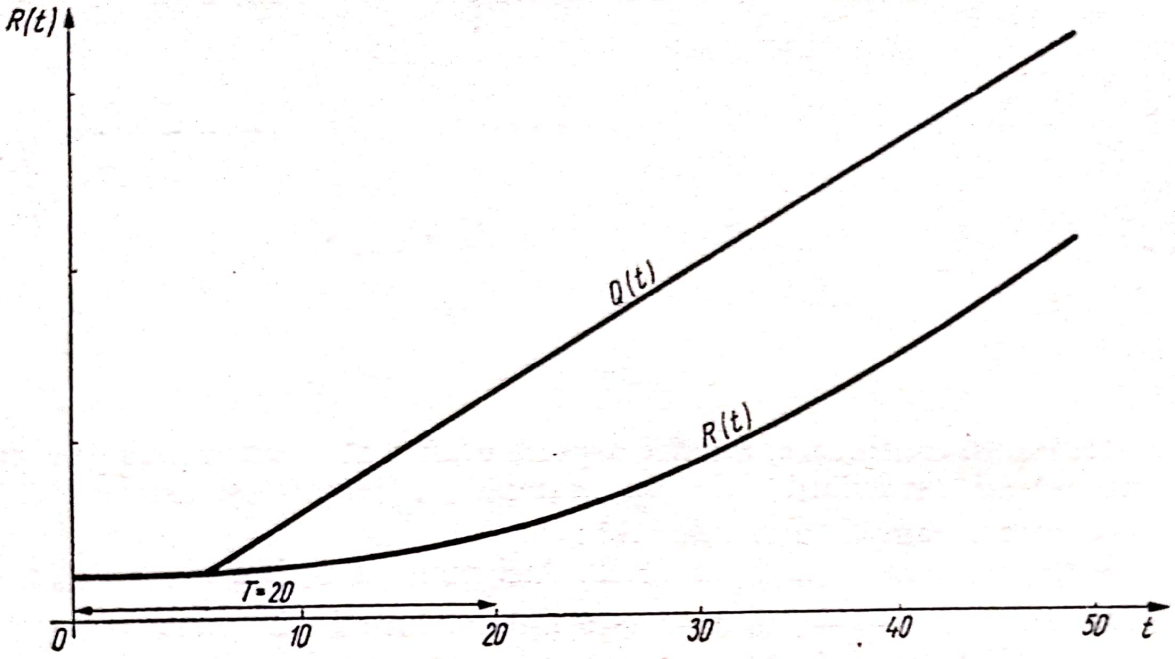


Fig. 8.11

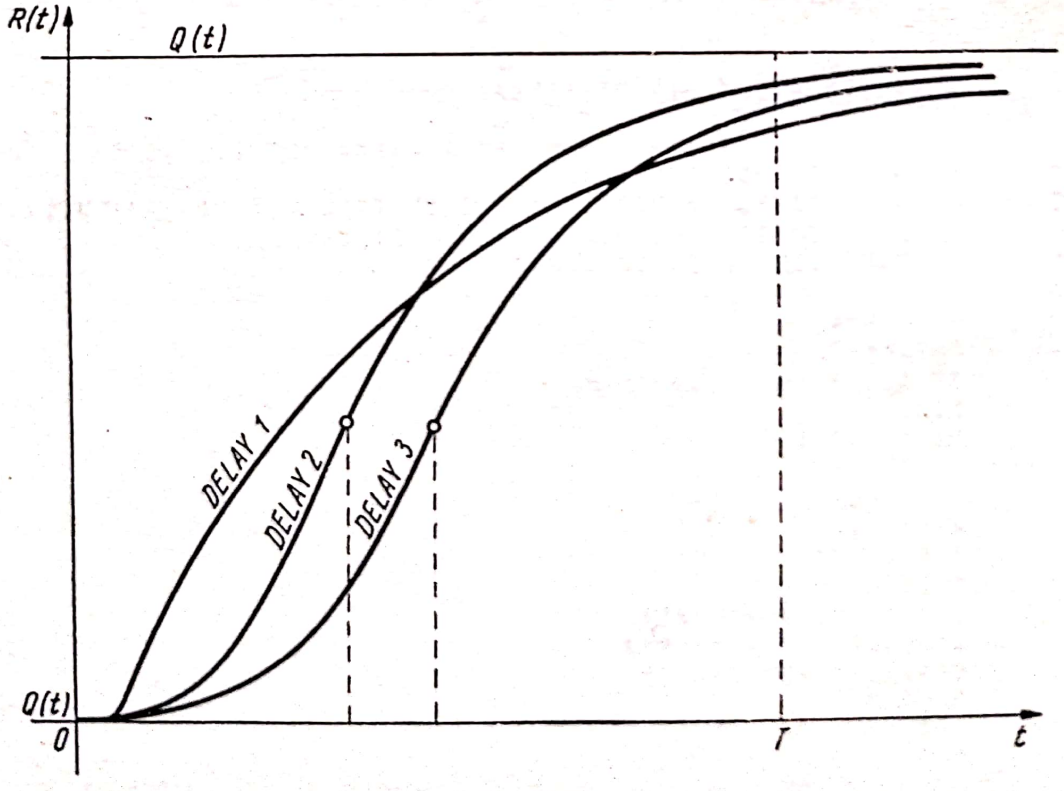


Fig. 8.12

Scanned with OKEN Scanner

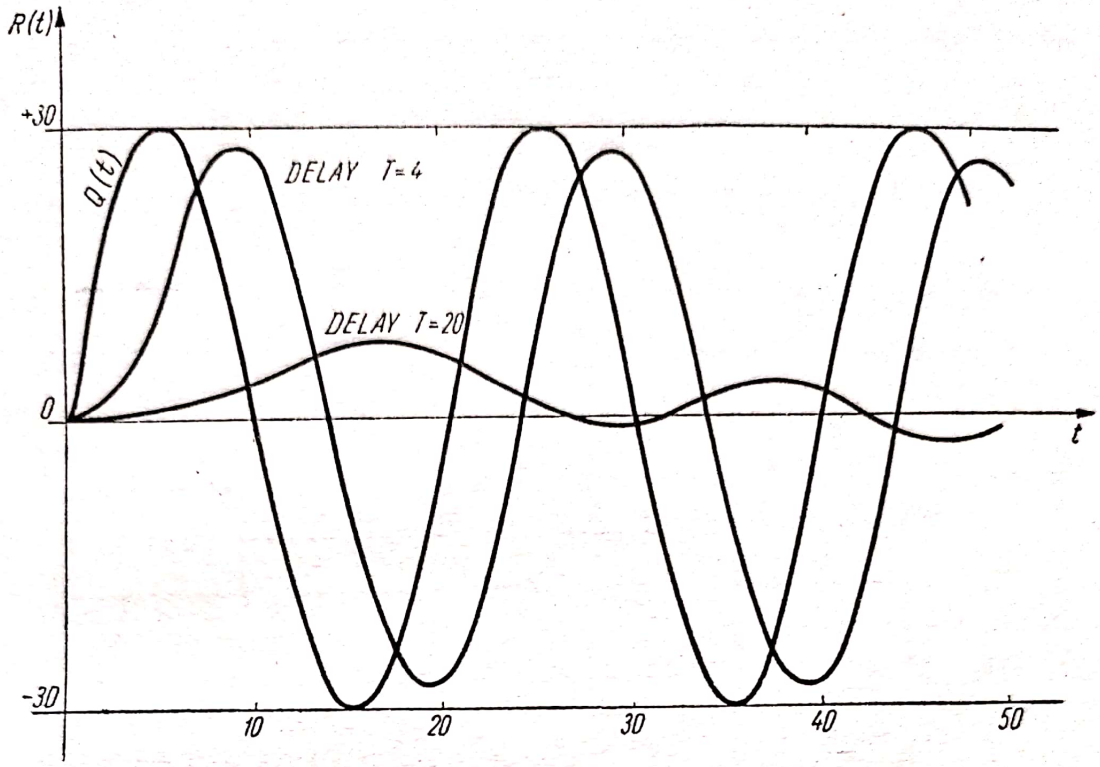


Fig. 8.13

unde:

TN este constanta de timp a nivelării exponențiale,
R — ritmul de nivelat și
RN — ritmul nivelat.

8.5. UN EXEMPLU DE APLICAȚIE AL METODEI FORRESTER

Vom prezenta un model dinamic, al unui sistem real, construit după metoda descrisă anterior.

Mai întîi vom formula modelul sub forma unor propoziții.

Dorim să cunoaștem evoluția viitoare a procesului de pompare precum și a cantităților de petrol pompate de la cîteva schele de extracție, la o rafinărie.

Intrarea petrolului în rezervoarele rafinăriei este condiționată în principal de doi factori:

1) Planul rafinăriei, dat, reprezentat de noi printr-o funcție tabelată *PLR*, care reprezintă în timp cantitățile de petrol de primit în rafinărie.

2) Cantitatea de petrol pompată de schelele de extracție, reprezentată de noi prin variabila de stare *SCHELA* (fig. 8.14).

Cantitatea de petrol pompată de schele depinde și ea de doi factori (v. fig. 8.14):

1° planul de extracție (funcția tabelată *PLS*);
2° cererea rafinăriei (variabile de stare *CERERE*).

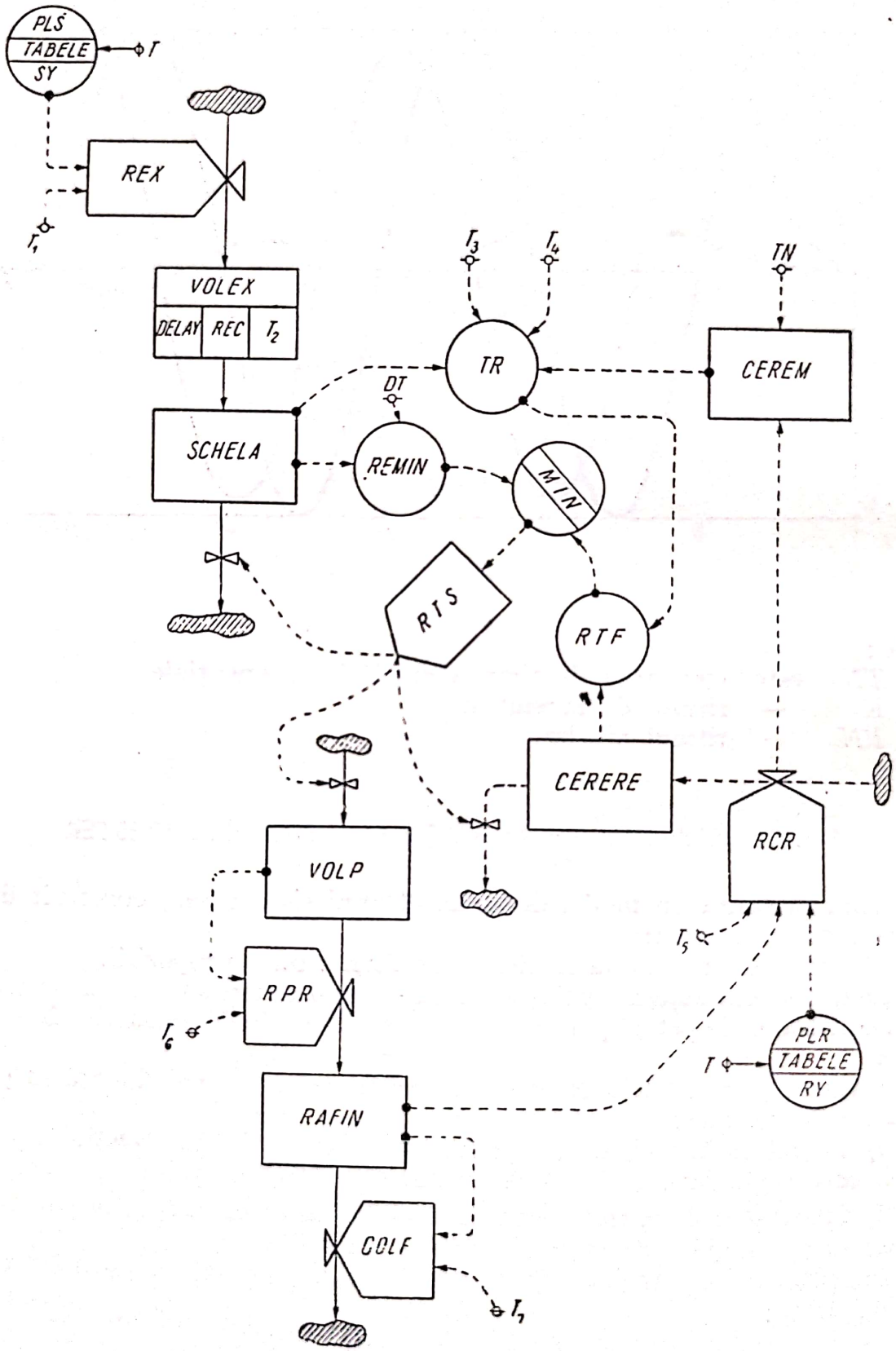


Fig. 8.14

Scanned with OKEN Scanner

Mai cunoaştem, din procesul fizic, timpii de lucru în diversele sectoare ale schelei şi rafinăriei, mărimi notate în figura 8.14 cu T_i ($i = 1, 2, \ldots, 7$). Cu aceste date şi la acest grad de generalitate vom construi modelul dinamic.

Modelul grafic (fig. 8.14) prezintă modul de interconectare al nivelelor şi ritmurilor care au fost luate în considerare în sistemul real.

Ecuaţiile modelului dinamic (scrise în limbajul Dynamo), sînt următoarele:

1) $REX.KL = PLS.K$

unde:

REX reprezintă ritmul de extracţie la schelă (el este modelat ca un element decizional tipic neliniar şi este dat de funcţia tabelată);

PLS — planul schelei, care reprezintă experienţa şi previziunea celor care lucrează în serviciile extracţie schelă;

$PLS.K = TABELE(SY,SX,6,T)$ unde *TABELE* este o funcţie FORTRAN de tabelare scrisă pentru utilizarea în modelele dinamice; ea are ca argument timpul (în lista 8.22 este prezentat programul acestei subrutine de tip funcţie);

2) $VOLEX.K = VOLEX.J + (DT)(REX.JK - REC.JK)$

unde *VOLEX* reprezintă volum extras, ca variabilă de nivel, adică este cantitatea de petrol brut care după extracţie este trimisă în rezervoarele de separare;

```
      DOUBLE PRECISION FUNCTION TABELE(FY,FX,N,TA)
C
C                SUBRUTINA DE INTERPOLARE A FUNCTIILOR TABELATE
C        FY      VALOAREA FUNCTIEI      -VECTOR
C        FX      VALOAREA ARGUMENTULUI  -VECTOR .
C        N       NUMARUL DE VALORI
C        TA      VALOAREA ARGUMENTULUI PENTRU CARE SE INTERPOLEAZA DBL
C
      DOUBLE PRECISION T,TA
      DIMENSION FY(N),FX(N)
      T=TA
      IF (T.GT.FX(1)) GO TO 1
      TABELE=FY(1)
      RETURN
    1 N1=N-1
      DO 2 I=1,N1
      IF (T-FX(I+1)) 4,3,2
    2 CONTINUE
      TABELE=FY(N)
      RETURN
    3 TABELE=FY(I+1)
      RETURN
    4 TABELE=FY(I)+(FY(I+1)-FY(I))/(FX(I+1)-FX(I))*(T-FX(I))
      RETURN
      END
```

Lista 22

Scanned with OKEN Scanner

3) $REC.KL = DELAY3(REX.JK,T2)$

unde *REC* este ritmul de separare a petrolului, modelat cu o întîrziere de ordinul III (cu timpul de întîrziere *T2*);

4) $SCHELA.K = SCHELA.J + (DT)(REC.JK - RTS.JK)$

unde *SCHELA* este variabila de nivel şi reprezintă volumul de petrol depozitat la schelă în rezervoarele de „curat";

5) $RTS.KL = MIN(RTF.KL,REMIN.KL)$

unde *RTS* este ritmul de transport de la schelă, fiind elementul decizional cheie al întregului model: condiţionează evoluţia variabilelor de stare *SCHELA*, *CERERE* şi *VOLP*;

6) $RTF.KL = CERERE.K/TR$

unde *RTF* este ritmul de transport figurat, care devine efectiv după confruntarea cu *REMIN* (are aspectul buclei de reacţie pozitive);

7) $TR = T3 + T4\left|\frac{CEREM.K}{SCHELA.K}\right|$

unde *TR* este timpul de transport de la schelă, mărime formată din doi termeni: *T3* — timpul normal de transport prin conducte şi *T4* — timpul de transport legat de rezervele schelei. *TR* face legătura între variabilele de nivel *CERERE* şi *SCHELA* prin variabila medie *CEREM*;

8) $CEREM.K = CEREM.J + (DT)/(TN)(RCR.JK - CEREM.J)$

unde *CEREM* este cererea medie a rafinăriei, o variabilă de nivel care reprezintă medierea exponenţială a ritmului de cerere al rafinăriei;

9) $REMIN.KL = SCHELA.K/DT$

unde *REMIN* este ritmul minim de pompare al schelei;

10) $CERERE.K = CERERE.J + (DT)(RCR.JK - RTS.JK)$

unde *CERERE*, ca variabilă de nivel, materializează sub formă informaţională cererea de petrol a rafinăriei;

11) $RCR.KL = (PLR.K - RAFIN.K)/T5$

unde: *RCR* este ritmul de cerere al rafinăriei — element decizional de ordinul II, iar *T5* este timpul în care se doreşte pomparea petrolului în rezervoarele rafinăriei pentru anularea diferenţei între cantităţile reale şi cele planificate;

12) $PLR.K = TABEL(PLR.K)$

unde *PLR* este planul rafinăriei (funcţie tabelată avînd ca argument timpul);

13) $VOLP.K = VOLP.J + (DT)(RTS.JK - RPR.JK)$

unde *VOLP* este volumul de primire, o variabilă de nivel ce reprezintă cantitatea de petrol în transport pe conducte;

Scanned with OKEN Scanner

```
C       LISTA 8.23
C
C       PROGRAM FORTRAN PENTRU SIMULARI PE
C       MODELUL DINAMIC AL EVOLUTIEI REZERVELOR DE PETROL
      DOUBLE PRECISION ANTET,T,TABELE
      DIMENSION ANTET(10),SCARA(10)
      DIMENSION BLOC(51,10),RY(4),RX(4),V(4),SY(4),SX(4)
      DATA NDIM/51/
      DATA TI,TF,DT/0.,50.,0.1/
      DATA T1/1/,T2/3/,T3/2/,T4/1/,T5/3/,T6/3/,T7/4/,C1/1.1/,N/4/,C2/2/
      DATA RY/500.,500.,600.,600./,RX/0.,12.,13.,35/
      DATA SY/000.,600.,700.,700./ SX/0.,20.,27.,36/
      DATA ANTET /'   VOLEX  SCHELA  CERERE    VOLP    RAFI..   CEREM
     *RTS'/
      DATA SCARA /1.,0/
      T=TI
      K=1
C
C     MOMENTUL DE MEMORARE
      KMAX=(TF-TI)/DT+1
      MM=1
   10 KMAX=KMAX-NDIM
      IF(KMAX)12,12,11
   11 MM=MM+1
      GO TO 10
   12 KR=0
C
C       VALORI INITIALE
      PLS=TABELE (SY,SX,4,T)
      VOLEX= PLS
      SCHELA=PLS*C2
      CERERE=TABELE(RY,RX,4,T)/C2
      VOLP=TABELE(RY,RX,4,T)/C2
      RAFIN=TABELE(RY,RX,4,T)*C1
      CEREM=TABELE(RY,RX,4,T)/C2/15
      GO TO 2
    1 CONTINUE
C
C     ECUATII DE NIVEL
      VOLEX=VOLEX+DT*(REX-REC)
      SCHELA=SCHELA+DT*(REC-RTS)
      CERERE=CERERE+DT*(RCR-RTS)
      CEREM=CEREM+DT/T4*(RCR-CEREM)
      VOLP =VOLP+DT*(RTS-RPR)
      RAFIN=RAFIN+DT*(RPR-GOLF)
    2 CONTINUE
C
C     ECUATII AUXILIARE
      PLS=TABELE (SY,SX,4,T)
      REMIN=SCHELA/DT
      TR=T5+T4*ABS(CEREM /SCHELA)
      RTF=CERERE/TR
      PLR=TABELE(RY,RX,4,T)
C
C     ECUATII DE RITM
      REX=PLS/T1
      REC=DELAY (REX,V,DT,12)
      RTS=AMIN1(REMIN,RTF)
      RCR=(PLR-RAFIN)/T5
      RPR=VOLP/T6
      GOLF=RAFIN/T7
C
C     SALVARE DE TABELARE
      IF (KR.NE.0) GO TO 15
      KR=MM
      BLOC(K,1) =VOLEX
      BLOC(K,2) =SCHELA
      BLOC(K,3) =CERERE
      BLOC(K,4) =VOLP
      BLOC(K,5) =RAFIN
      BLOC(K,6) =CEREM
      BLOC(K,7) =RTS
      K=K+1
   15 KR=KR-1
      T=T+DT
      IF(K.LE.NDIM) GO TO 1
      CALL PRNT (BLOC,NDIM,7,SCARA,TI,DT,MM,ANTET,10)
      STOP
      END
```

Lista 8.23

Scanned with OKEN Scanner

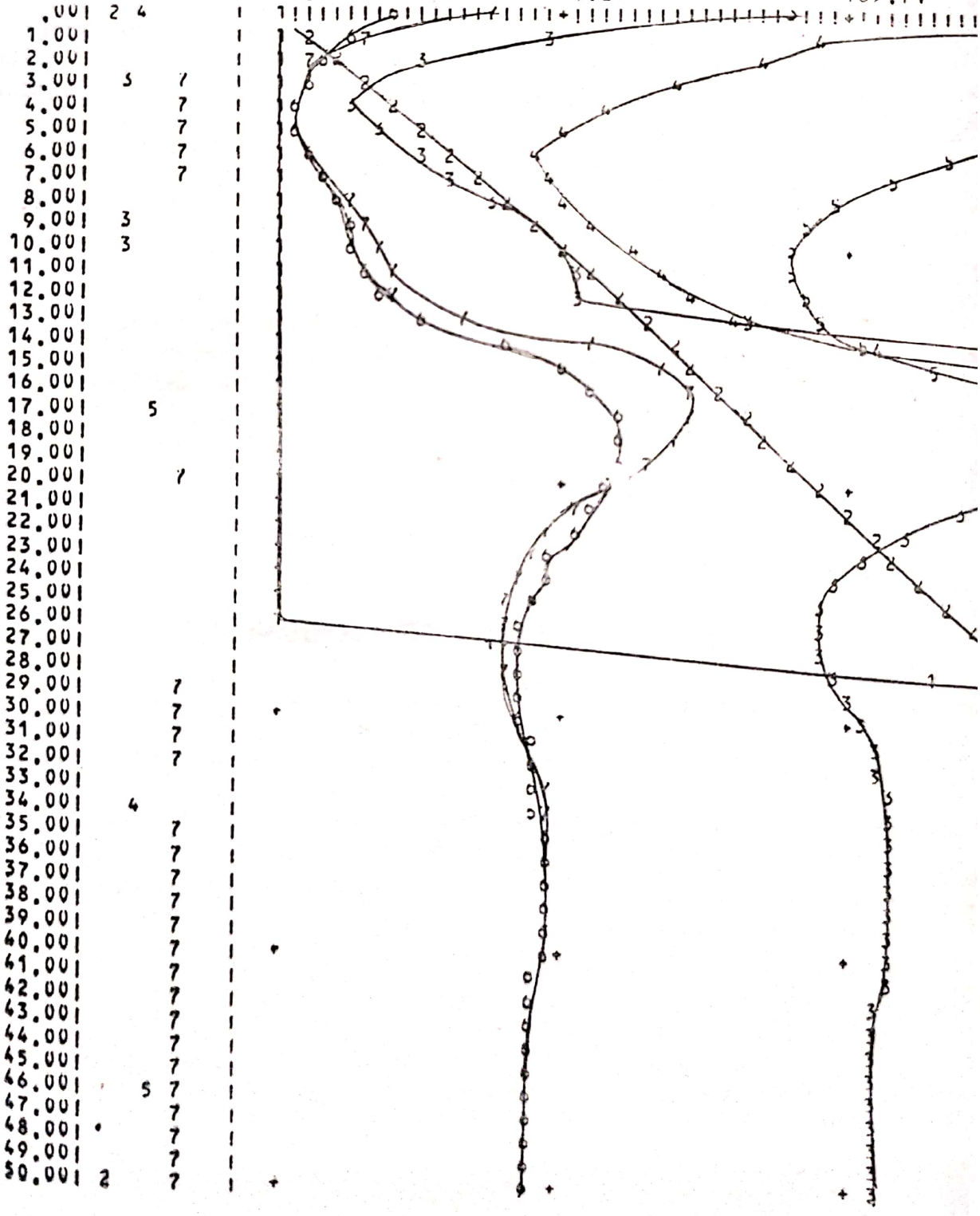
T	VOLEX	SCHELA	CERERE	VULP	RAFIN	CEREM	RTS
.00	600.00	1200.0	150.00	150.00	330.00	50.000	73.469
1.00	600.00	1741.9	86.306	156.19	302.64	37.598	42.692
2.00	600.00	2307.3	54.246	140.78	279.64	29.804	26.949
3.00	600.00	2883.9	41.088	120.35	256.21	25.472	20.454
4.00	600.00	3464.1	39.539	102.76	232.20	23.883	19.702
5.00	600.00	4043.4	44.945	91.139	209.11	24.418	22.405
6.00	600.00	4619.1	54.052	86.048	188.65	26.472	26.949
7.00	600.00	5189.8	64.497	86.686	172.08	29.462	32.157
8.00	600.00	5755.3	74.605	91.613	160.05	32.861	37.196
9.00	600.00	6316.1	83.279	99.205	152.59	36.235	41.520
10.00	600.00	6873.0	89.919	107.94	149.26	39.261	44.831
11.00	600.00	7427.1	94.327	116.56	149.29	41.735	47.032
12.00	600.00	7979.5	96.618	124.16	151.76	43.562	48.177
13.00	600.00	8525.6	136.61	135.34	156.04	55.266	68.086
14.00	600.00	9042.0	199.74	169.35	165.76	75.719	99.452
15.00	600.00	9533.9	253.78	214.26	185.44	90.551	116.34
16.00	600.00	10014.	247.75	256.46	213.96	100.23	123.26
17.00	600.00	10490.	247.76	289.45	247.49	105.44	123.26
18.00	600.00	10969.	238.58	310.92	281.76	106.92	118.71
19.00	600.00	11453.	224.19	321.11	313.16	105.61	111.58
20.00	600.00	11945.	207.78	321.70	339.17	102.42	103.45
21.00	600.00	12445.	191.81	315.12	358.47	98.236	95.528
22.00	600.00	12953.	177.96	303.98	370.78	93.797	88.659
23.00	600.00	13467.	167.20	290.73	376.67	89.663	83.322
24.00	600.00	13985.	159.88	277.39	377.23	86.210	79.695
25.00	600.00	14506.	155.88	265.45	373.86	83.635	77.715
26.00	600.00	15029.	154.73	255.84	367.99	81.983	77.156
27.00	644.80	15552.	155.79	248.99	360.94	81.188	77.692
28.00	737.71	16081.	158.33	244.90	353.83	81.104	78.968
29.00	808.08	16631.	161.68	243.27	347.44	81.548	80.641
30.00	852.22	17205.	165.23	243.62	342.28	82.328	82.420
31.00	876.67	17798.	168.55	245.36	338.59	83.267	84.079
32.00	889.12	18400.	171.33	247.94	336.38	84.219	85.468
33.00	895.11	19008.	173.40	250.83	335.49	85.075	86.507
34.00	897.86	19619.	174.73	253.63	335.65	85.767	87.176
35.00	899.08	20230.	175.38	256.06	336.57	86.264	87.503
36.00	899.61	20842.	175.45	257.95	337.93	86.564	87.544
37.00	899.84	21455.	175.10	259.24	339.47	86.688	87.372
38.00	899.93	22067.	174.47	259.94	340.96	86.669	87.065
39.00	899.97	22680.	173.71	260.15	342.26	86.549	86.692
40.00	899.99	23294.	172.95	259.97	343.28	86.369	86.313
41.00	899.99	23908.	172.25	259.53	343.97	86.165	85.972
42.00	900.00	24522.	171.69	258.95	344.35	85.966	85.695
43.00	900.00	25136.	171.28	258.32	344.47	85.793	85.496
44.00	900.00	25751.	171.03	257.74	344.38	85.657	85.375
45.00	900.00	26365.	170.93	257.25	344.14	85.564	85.325
46.00	900.01	26980.	170.93	256.88	343.83	85.512	85.332
47.00	900.01	27595.	171.02	256.65	343.50	85.496	85.379
48.00	900.01	28209.	171.16	256.55	343.19	85.508	85.451
49.00	900.01	28824.	171.32	256.51	342.93	85.539	85.532
50.00	900.02	29438.	171.47	256.57	342.73	85.581	85.612

Lista 8.24

Scanned with OKEN Scanner

GRAFIC:1	VOLEX	VALOAREA MAXIMA	900.02	PENTRU T	50.00
GRAFIC:2	SCHELA	VALOAREA MAXIMA	29438.	PENTRU T	50.00
GRAFIC:3	CERERE	VALOAREA MAXIMA	247.76	PENTRU T	17.00
GRAFIC:4	VOLP	VALOAREA MAXIMA	321.70	PENTRU T	20.00
GRAFIC:5	RAFIN	VALOAREA MAXIMA	377.23	PENTRU T	24.00
GRAFIC:6	CEREM	VALOAREA MAXIMA	106.92	PENTRU T	18.00
GRAFIC:7	RTS	VALOAREA MAXIMA	123.26	PENTRU T	16.00

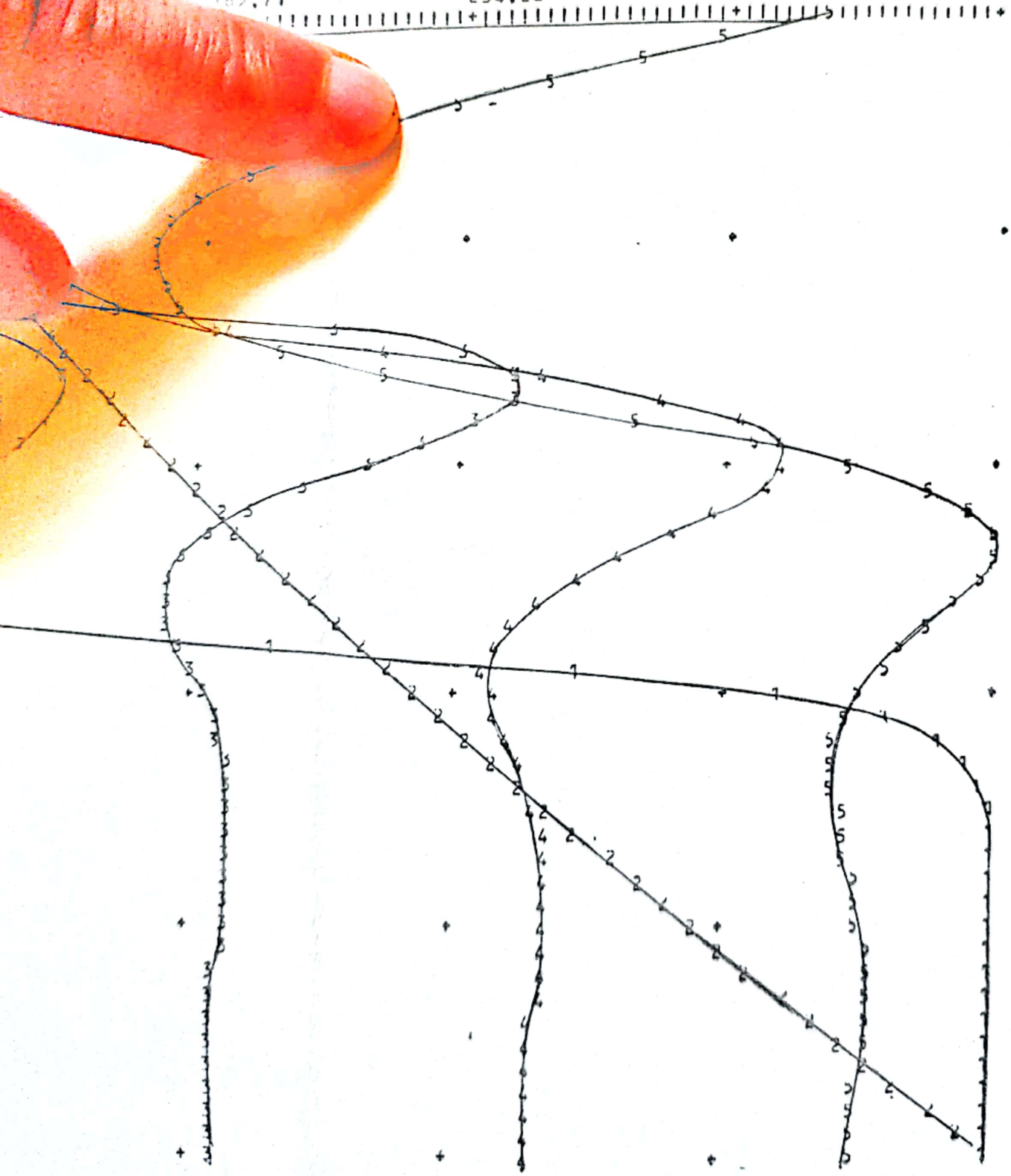
SCARA			
.	600.00	660.00	720.01
2	1200.0	6847.6	12495.
3	19.702	91.208	162.71
4	19.702	91.208	162.71
5	19.702	91.208	162.71
6	19.702	91.208	162.71
7	19.702	91.208	162.71



Lista 8.25

00.02	PENTRU T	50.00	VALOAREA MINIMA	600.00	PENTRU T	.0000
9438.	PENTRU T	50.00	VALOAREA MINIMA	1200.0	PENTRU T	.0000
47.70	PENTRU T	17.00	VALOAREA MINIMA	39.539	PENTRU T	4.000
21.70	PENTRU T	20.00	VALOAREA MINIMA	86.048	PENTRU T	6.000
77.23	PENTRU T	24.00	VALOAREA MINIMA	149.26	PENTRU T	10.00
6.92	PENTRU T	18.00	VALOAREA MINIMA	23.883	PENTRU T	4.000
3.20	PENTRU T	16.00	VALOAREA MINIMA	19.702	PENTRU T	4.000

0.00	720.01	780.01	840.01	900.02
7.6	12495.	18143.	23791.	29438.
08	162.71	234.22	305.73	377.23
	162.71	234.22	305.73	377.23
	162.71	234.22	305.73	377.23
	162.71	234.22	305.73	377.23
	162.71	234.22	305.73	377.23



Lista 8.25

Scanned with OKEN Scanner

14) $RPR.KL = VOLP.K/T6$

unde: RPR este ritmul de primire în rezervoarele rafinăriei şi $T6$ este timpul de transport pe conducte;

15) $RAFIN.K = RAFIN.J + (DT)(RPR.JK - GOLF.JK)$

unde $RAFIN$ reprezintă rezervoarele rafinăriei pregătite să primească petrolul schelelor (o variabilă de nivel);

16) $GOLF.JK = RAFIN.K/T7$

unde: $GOLF$ reprezintă ritmul de golire a rezervoarelor spre procesele de prelucrare, iar $T7$ este timpul de golire.

Prezentăm în lista 8.23 programul FORTRAN pentru simulări pe modelul dinamic al exemplului considerat; rezultatele uneia dintre simulările făcute cu modelul de mai sus sînt redate în lista 8.24 (în formă tabelată) şi în lista 8.25 (sub formă de grafice).

Interpretarea rezultatelor din tabele şi din grafice conduce la următoarele concluzii:

1° Rezervoarele rafinăriei sînt corect alimentate prin sistemul ales. O intrare treaptă a variabilei PLR în ziua a treisprezecea cu o creştere de 100% provoacă o stabilizare rapidă în ziua a patruzecea a cantităţilor din rezervorul $RAFIN$ la valorile nominale planificate. Această intrare treaptă a fost aleasă tocmai pentru a pune în evidenţă răspunsul sistemului, adică sensibilitatea şi stabilitatea lui.

2° Rezervoarele schelei pot face faţă şi altor cereri din afara sistemului. Astfel, variabila de nivel $SCHELA$ prezintă o creştere aproape liniară între timpul iniţial şi final. Se impune deci conectarea rezervoarelor schelei şi la alte conducte de consumatori.

3° Variaţia variabilei de nivel $CERERE$ este corectă, ea „balansînd" surplusurile şi lipsurile din rezervoarele rafinăriei pentru a aduce nivelul petrolului la valorile planificate.

8.6. ALGORITM DE CONDUCERE OPTIMALĂ*

Din cele arătate în paragrafele anterioare a reieşit că realizarea dezideratului de a conduce în timp un sistem real depinde de obţinerea ecuaţiilor de stare ale sistemului — respectiv modelul dinamic — şi apoi de aplicarea unui algoritm de optimizare adecvat.

Întrucît modelul dinamic a fost realizat, urmează să scriem imediat ecuaţiile de stare (ne referim în continuare la aplicaţia de la § 8.5).

Variabilele de stare sînt:

$$\begin{aligned} x_1 &- VOLEX \\ x_2 &- SCHELA \\ x_3 &- CERERE \end{aligned} \qquad (8.40)$$

* Pentru aprofundarea noţiunilor prezentate în cadrul acestui subcapitol, propunem cititorului să consulte lucrarea *Sisteme informatice, modele ale conducerii şi sisteme conduse* de P. Constantinescu şi C. V. Negoiţă, apărută în 1975 la Editura tehnică Bucureşti.

Scanned with OKEN Scanner

$$x_4 - VOLP$$
$$x_5 - RAFIN$$
$$x_6 - CEREM$$

Alegem următoarele variabile de comandă:

$$\begin{aligned} u_1 &- T2 \\ u_2 &- T3 \\ u_3 &- T5 \\ u_4 &- T6 \\ u_5 &- TN \end{aligned} \qquad (8-41)$$

și de asemenea, pentru fiecare comandă, intervalul de variație propriu $u_i \in [a_i, b_i]$ și comanda nominală $u(0)$.

Vom formula următoarea cerință pentru model: „ritmul de transport $RT'S$ să fie maxim la momentul de timp final".

Ecuația sa este:

$$RTS = MIN(RTF, REMIN)$$

Cu notațiile anterioare vom putea scrie criteriul de optim:

$$I(t) = \{min[(x_2/u_3),\ (u_2 + T4 \cdot x_6/x_2)]\}. \qquad (8-42)$$

Pentru momentul final t_f cerem:

$$I(t_f) = max\{min[(x_2/T3),\ (T2 + T4 \cdot x_6/x_2)]\}. \qquad (8-43)$$

Pentru a realiza cerințele de mai sus vom folosi următorul algoritm de conducere optimală:

— fie un sistem de ecuații:

$$\begin{aligned} \dot{x} &= f[\mathbf{x}(t),\ \mathbf{u}(t)] \text{ cu vectorii } \mathbf{x}(t) \in R^n \\ x(0) &= x_0 \qquad\qquad\quad \mathbf{u}(t) \in R^m \end{aligned} \qquad (8-44)$$

și pentru care există restricțiile $\mathbf{u}(t) \in [a, b]$;

— atașăm sistemului de mai sus o funcțională de performanță $I = \varphi\ [\mathbf{x}(t_f)]$ pe care dorim să o maximizăm la momentul de timp final.

Datorită faptului că atît sistemul de ecuații cît și criteriul de optim sînt funcții neliniare, nu ne putem propune să aflăm optimul în mod direct, ci vom pleca dintr-un punct $\mathbf{x}_{nom}$ din spațiul stărilor și vom căuta să ne apropiem de optim prin pași succesivi maximizînd la fiecare pas funcționala de performanță I. Vom comuta în acest fel o problemă de optim neliniară într-o problemă liniară pentru care există soluții.

Vom porni cu valorile nominale ale stărilor și comenzilor respective $\mathbf{x}_{nom}$ și $\mathbf{u}_{nom}$ și vom da creșteri mici comenzilor, presupunînd că la creșteri mici pentru comenzi corespund creșteri mici pentru stările sistemului:

$$\begin{aligned} \hat{\delta}\mathbf{x}(t) &= \mathbf{x}(t) - \mathbf{x}_{nom}(t), \\ \hat{\delta}\mathbf{u}(t) &= \mathbf{u}(t) - \mathbf{u}_{nom}(t). \end{aligned} \qquad (8-45)$$

Scanned with OKEN Scanner

Folosind (8—44) şi (8—45) putem scrie:

$$\hat{\delta}\dot{x}(t) = \dot{x}(t) - \dot{x}_{nom}(t) = f[\mathbf{x}_{nom}(t) + \hat{\delta}\mathbf{x}(t),\ \mathbf{u}_{nom}(t) + \hat{\delta}\mathbf{u}(t)] - f[\mathbf{x}_{nom}(t), \mathbf{u}_{nom}(t)]. \tag{8—46}$$

Dezvoltînd în serie Taylor şi neglijînd termenii de ordin superior scriem relaţiile:

$$\hat{\delta}\dot{x}(t) = \frac{\partial f}{\partial x}[\mathbf{x}_{nom}(t),\ \mathbf{u}_{nom}(t)]\hat{\delta}\mathbf{x}(t) + \frac{\partial f}{\partial u}[\mathbf{x}_{nom}(t),\ \mathbf{u}_{nom}(t)] \cdot \hat{\delta}\mathbf{u}(t). \tag{8—47}$$

Notăm:

$$\frac{\partial f}{\partial x}[\mathbf{x}_{nom}(t),\ \mathbf{u}_{nom}(t)] = \mathbf{A}(t)$$

şi

$$\frac{\partial f}{\partial u}[\mathbf{x}_{nom}(t),\ \mathbf{u}_{nom}(t)] = \mathbf{B}(t),$$

respectiv matricele iacobine ale stărilor şi comenzilor.

Obţinem sistemul liniar:

$$\hat{\delta}\dot{x}(t) = \mathbf{A}(t) \cdot \hat{\delta}\mathbf{x}(t) + \mathbf{B}(t) \cdot \hat{\delta}\mathbf{u}(t). \tag{8—48}$$

Acestui sistem îi asociem conform principiului maximului sistemul adjunct:

$$\begin{cases} \dot{\psi}(t) = -\psi(t) \cdot \mathbf{A}(t) \\ \psi(t_f) = \mathbf{C} \end{cases} \tag{8—49}$$

unde:

$$\mathbf{C} = \frac{\partial \Phi}{\partial x}[\mathbf{x}_{nom}(t_f)] \tag{8—50}$$

este iacobianul $\mathbf{C}$ care cuprinde valorile derivatelor parţiale ale funcţiei obiectiv la momentul de timp final.

Variaţia dată comenzilor va produce o variaţie şi la funcţională de performanţă. Corecţia corespunzătoare la funcţională este:

$$\hat{\delta}\Phi[\mathbf{x}(t_f)] = \Phi[\mathbf{x}(t_f)] - \Phi[\mathbf{x}_{nom}(t_f)] = \Phi[\mathbf{x}_{nom}(t_f) + \hat{\delta}\mathbf{x}(t_f)] - \Phi[\mathbf{x}_{nom}(t_f)]. \tag{8—51}$$

Dezvoltînd la fel, în serie Taylor şi neglijînd termenii mici, de ordin superior

$$\hat{\delta}\Phi[\mathbf{x}(t_f)] = \frac{\partial \Phi}{\partial x}[\mathbf{x}_{nom}(t_f)] \cdot \hat{\delta}\mathbf{x}(t_f) \tag{8—52}$$

şi avînd în vedere (8—49) putem scrie:

$$\hat{\delta}\Phi[\mathbf{x}(t_f)] = \psi(t_i) \cdot \hat{\delta}\mathbf{x}(t_f) - \psi(t_0) \cdot \hat{\delta}\mathbf{x}(t_0), \text{ cu } \hat{\delta}\mathbf{x}(t_0) = 0;$$

Scanned with OKEN Scanner

$$\hat{\delta}\Phi[\mathbf{x}(t_f)] = \int_0^{t_f} \frac{\mathrm{d}}{\mathrm{d}t}(\psi \cdot \hat{\delta}\mathbf{x}) = \int_0^{t_f} (\mathring{\psi}\hat{\delta}\mathbf{x} + \psi\hat{\delta}\mathring{\mathbf{x}})\,\mathrm{d}t;$$

$$\hat{\delta}\Phi[\mathbf{x}(t_f)] = \int_0^{t_f} [\mathring{\psi}(t) \cdot \mathbf{B}(t) \cdot \hat{\delta}\mathbf{u}(t)]\,\mathrm{d}t = \int_0^{t_f} [\mathbf{G}(t) \cdot \hat{\delta}\mathbf{u}(t)]\,\mathrm{d}t, \quad (8-53)$$

cu notația:

$$\mathbf{G}(t) = [\psi(t) \cdot \mathbf{B}(t)]. \quad (8-54)$$

Rezultă că $\Phi[\mathbf{x}(t_f)] > \Phi[\mathbf{x}_{nom}(t_f)]$ dacă $\hat{\delta}\,\Phi\,[\mathbf{x}(t_f)]$ va avea același semn cu $\mathbf{G}(t)$ și creșterea este maximă cînd alegem $\hat{\delta}\mathbf{u}(t) = h\mathbf{G}(t)$.

(Conform inegalității lui Schwartz: $<\mathbf{x} \cdot \mathbf{y}> \leqslant \|\mathbf{x}\| \cdot \|\mathbf{y}\|$, cînd cele două mărimi $\hat{\delta}\mathbf{u}(t)$ și $\mathbf{G}(t)$ sînt liniar dependente produsul lor este maxim.)

Constanta h se alege în așa fel încît să nu ieșim din intervalul admis pentru comenzi.

Pentru ca să rămînem în intervalul admis al comenzilor adică:

$$\mathbf{u}(t) + \hat{\delta}\mathbf{u}(t) \in [a,\ b], \quad (8-55)$$

vom calcula:

$$h_{max} = \begin{cases} \min\limits_{i,t} \dfrac{b_i - u_i(t)}{G_i(t)} \text{ dacă } G_i(t) > 0 \\ \min\limits_{i,t} \dfrac{u_i(t) - a_i}{G_i(t)} \text{ dacă } G\,(t) < 0 \end{cases} \quad (8-56)$$

Pentru $G_i(t) = 0$ vom pune $h_{max} = \infty$ *calculator.*

Neputînd afla în mod direct (din relațiile de pînă acum) mărimea h, vom căuta să calculăm printr-o metodă numerică o altă mărime h^* care se apropie cît mai mult de h maximizînd $\hat{\delta}\mathbf{u}(t)$. Criteriul de alegere a lui h^* va fi, evident, maximizarea lui $\hat{\delta}\Phi[\mathbf{x}(t_f)]$. Astfel:

$$h^*(t) = \frac{p}{q}\, h_{max}(t), \quad (8-57)$$

unde p și q sînt două numere care aparțin șirului lui Fibonacci.

În relația de mai sus, nu ținem cont de condiția de liniaritate impusă de teorema lui Schwartz pentru R, ci vom căuta la fiecare moment de timp acel $h^*(t)$ care maximizează funcționala de performanță, ceea ce se poate admite pentru variații mici ale comenzilor în jurul valorilor nominale.

Înainte de a expune procedeul vom face o mică digresiune:

În anul 1902 Leonardo de Pisa, supranumit Fibonacci (filio Bonacci), unul dintre cei mai mari matematicieni ai evului mediu, a enunțat celebrul șir care îi poartă numele și care astăzi se întrebuințează la foarte mulți algoritmi de maximizare, interclasare etc.

În șirul lui Fibonacci fiecare număr este suma celorlalte două care îl precede astfel:

$$F_{n+2} = F_{n+1} + F_n \text{ cu } n \geqslant 0. \quad (8-58)$$

Scanned with OKEN Scanner

Pornind de la 0 obţinem următoarea secvenţă:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1587, ... (8—59)

Şirul, deşi pare a avea la început o creştere lentă, totuşi ajunge la termenul al douăzecelea la valori mai mari decît $4 \cdot 10^3$ (Fibonacci a propus calculul termenilor şirului ca un util exerciţiu de adunare!).

Şirul lui Fibonacci este de asemenea în legătură şi cu faimosul raport Φ care defineşte „proporţia divină" sau „numărul de aur" utilizat în arhitectura, sculptura şi pictura antică şi la fel în arta Renaşterii (Φ de la iniţiala lui Phidias, cel mai mare sculptor şi arhitect al antichităţii).

Astfel, se poate arăta relativ uşor că:

$$\gamma = \frac{1}{\Phi} = \lim_{n\to\infty} \frac{F_{n-1}}{F_n} = \frac{1}{2}(1 + \sqrt{5}) \qquad (8—60)$$

$$\gamma = 1{,}618, \text{ iar } \Phi = 0{,}618.$$

Euclid a cunoscut acest număr. El l-a numit „raportul mediei şi al extremelor":

$$\gamma = \frac{A}{B} = \frac{A+B}{B}. \qquad (8—61)$$

În ceea ce priveşte algoritmul de la care am pornit, noi vom utiliza şirul lui Fibonacci pentru a găsi pe segmentul h_{max} acel punct care determină maximul funcţionalei de performanţă. Acel punct va determina segmentul $h^* = h_{max}\frac{p}{q}$. Numărul q se alege în şir după precizia cerută pentru h^*.

Procedeul de lucru este următorul:

Se împarte segmentul h_{max} în raportul $\frac{F_{n-1}}{F_n}$ şi se fixează două puncte pornind de la cele două capete ale segmentului la distanţa $d_1 = \frac{F_{n-1}}{F_n} h_{max}$. Se calculează creşterile funcţionalei de performanţă şi se alege unul din cele două puncte pentru comparările ulterioare. Se calculează $d_2 = \frac{F_{n-2}}{F_n} h_{max}$ şi se aşază această lungime peste lungimea d_1 în care s-a găsit punctul de maxim, obţinîndu-se în continuare al doilea punct de comparat. Procedeul continuă la fel pînă la:

$$d_n = \frac{1}{F_n} \qquad \left(\text{în general } d_k = \frac{F_{n-k+1}}{F_n}\right). \qquad (8—62)$$

Aceste operaţii sînt rapid convergente către valoarea segmentului h^*, care produce maximul funcţiei de performanţă.

Trebuie, de asemenea, ţinut cont că deoarece am liniarizat ecuaţiile (8—43) procedura este fundamentată numai pentru valori mici ale comenzilor în jurul valorilor nominale.

Repetînd iterativ procedeul vom obţine creşteri succesive ale funcţionalei de performanţă.

În figura 8.15 prezentăm schema de calcul corespunzătoare acestui algoritm.

Scanned with OKEN Scanner

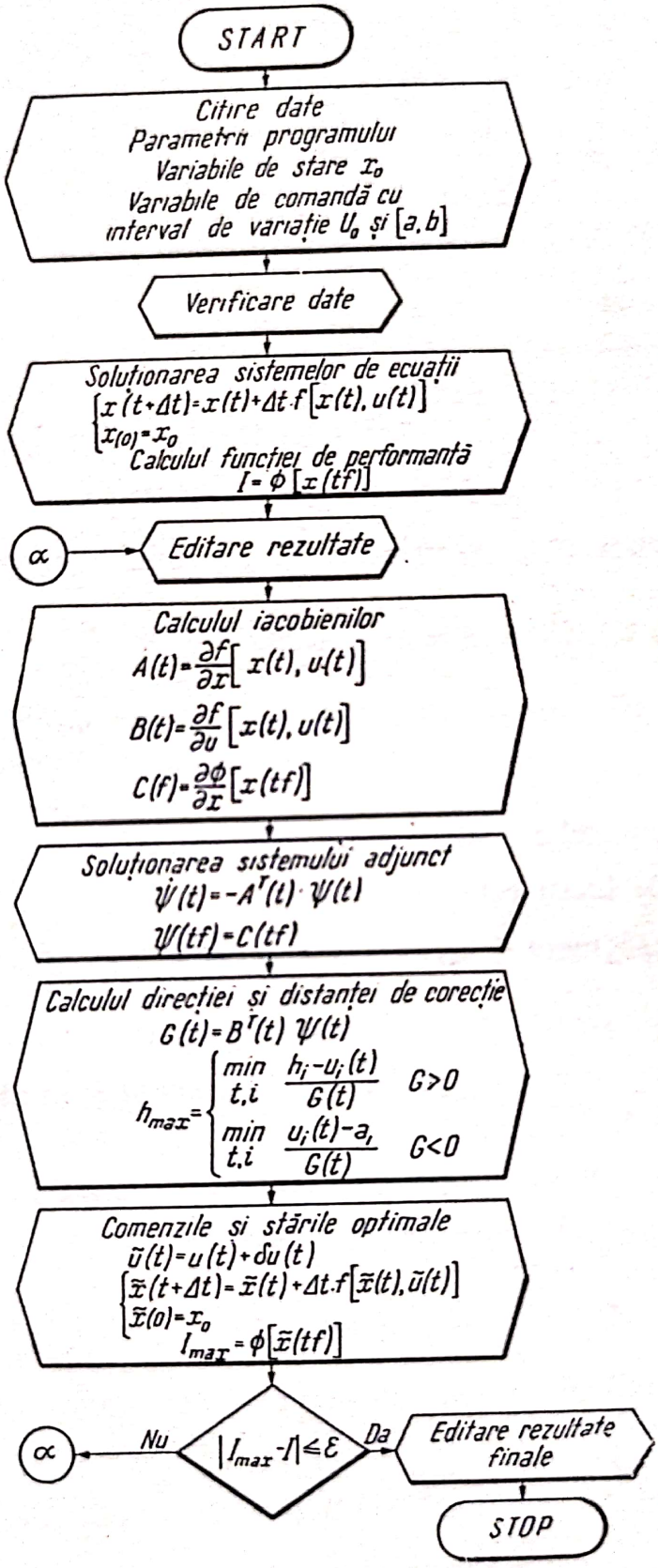


Fig. 8.15

Scanned with OKEN Scanner

T	X 1	X 2	X 3	X 4	X 5	X 6
0.00	600.00	1200.0	150.00	150.00	330.00	50.000
1.00	510.93	1831.0	86.293	156.20	302.64	37.598
2.00	426.09	2483.7	57.914	138.88	279.11	30.424
3.00	368.11	3116.9	42.541	105.00	269.87	25.743
4.00	334.20	3736.2	41.219	71.255	252.83	22.951
5.00	316.29	4341.2	51.547	56.073	221.02	23.288
6.00	307.46	4928.2	72.557	60.367	187.21	28.034
7.00	303.32	5507.7	122.60	68.835	159.70	38.540
8.00	301.45	6062.3	178.39	95.732	142.31	52.635
9.00	300.63	6607.9	215.77	124.02	134.35	61.862
10.00	300.28	7125.4	239.73	160.17	146.29	72.440
11.00	300.14	7641.5	229.88	191.03	160.95	73.371
12.00	300.08	8153.4	206.99	231.48	167.04	71.844
13.00	300.06	8647.9	253.83	260.24	198.50	89.671
14.00	300.05	9097.8	307.91	319.90	235.29	115.50
15.00	300.05	9558.5	330.96	353.71	277.64	126.22
16.00	300.05	10007.	319.73	383.05	324.58	129.92
17.00	300.05	10471.	265.18	342.09	413.15	120.42
18.00	300.04	10955.	199.62	309.98	449.36	104.76
19.00	300.04	11465.	160.06	278.99	457.88	92.560
20.00	300.04	11991.	136.89	262.11	436.46	83.203
21.00	300.04	12526.	128.78	243.50	414.34	77.558
22.00	300.04	13061.	129.86	228.84	392.09	74.838
23.00	300.04	13595.	136.31	220.08	371.25	74.326
24.00	300.04	14137.	154.21	182.38	374.03	74.693
25.00	300.04	14674.	167.48	183.82	345.56	75.055
26.00	300.04	15212.	192.25	188.87	319.94	77.681
27.00	344.84	15719.	256.07	227.51	297.32	95.357
28.00	437.75	16213.	285.78	267.37	296.89	106.24
29.00	508.11	16720.	306.44	322.63	291.25	113.93
30.00	552.25	17244.	336.97	366.11	306.81	125.85
31.00	542.59	17791.	359.15	413.45	340.41	138.70
32.00	557.73	18343.	356.61	432.03	366.27	137.79
33.00	572.65	18867.	350.50	460.49	403.30	143.10
34.00	550.33	19453.	279.13	470.70	427.22	127.51
35.00	564.46	20016.	242.61	451.13	460.03	119.16
36.00	578.28	20597.	194.84	384.81	510.61	106.09
37.00	588.15	21203.	141.06	345.91	506.06	89.114
38.00	594.04	21835.	109.17	281.37	506.46	76.061
39.00	597.18	22481.	94.932	243.86	471.70	62.131
40.00	598.74	23129.	101.91	217.05	434.86	60.886
41.00	599.48	23776.	109.95	199.91	399.68	60.810
42.00	599.81	24418.	123.68	192.48	368.62	63.068
43.00	599.96	25059.	145.29	175.39	355.97	66.530
44.00	600.02	25695.	166.00	170.67	336.98	70.223
45.00	600.05	26327.	184.84	169.10	324.82	73.880
46.00	600.06	26938.	207.91	197.96	305.94	82.619
47.00	600.07	27563.	252.69	213.99	290.36	90.830
48.00	600.08	28161.	370.68	244.24	289.98	118.[illegible]
49.00	600.08	28762.	436.04	262.77	296.40	128.37

Lista 8.26

Scanned with OKEN Scanner

T	U 1	U 2	U 3	U 4	U 5	OBIECTIV
.00	3.0000	2.0000	3.0000	3.0000	4.0000	69.481
1.00	3.0000	2.0000	3.0000	3.0000	4.0000	40.575
2.00	3.0000	2.0000	3.0000	3.0000	4.0000	27.704
3.00	3.0000	2.0000	3.0000	3.0000	4.0000	20.631
4.00	3.0000	2.0000	3.0000	3.0000	4.0000	20.308
5.00	3.0000	5.0141	1.0000	3.0000	3.7090	29.127
6.00	3.0000	2.0430	1.0000	3.0000	3.9990	40.026
7.00	3.0000	6.0000	1.3443	3.0000	4.0005	65.263
8.00	3.0000	6.0000	3.0051	3.0000	4.0011	89.947
9.00	3.0000	2.0000	3.0000	3.0000	4.0000	104.79
10.00	3.0000	2.0000	3.0000	3.0000	4.0000	115.88
11.00	3.0000	2.0000	3.0000	3.0000	4.0000	111.01
12.00	3.0000	2.0000	3.0000	3.0000	4.0000	100.12
13.00	3.0000	2.0000	3.0000	3.0000	4.0000	126.63
14.00	3.0000	2.0000	3.0000	3.0000	4.0000	151.42
15.00	3.0000	2.0000	3.0000	3.0000	4.0000	161.58
16.00	3.0000	2.0000	3.0000	3.0000	4.0000	155.52
17.00	3.0000	2.0001	3.0000	1.0000	4.0005	128.39
18.00	3.0000	2.0000	3.0000	3.0000	4.0000	96.899
19.00	3.0000	2.0000	3.0000	3.0000	4.0000	78.102
20.00	3.0000	2.0000	3.0000	3.0000	4.0000	67.527
21.00	3.0000	2.0000	3.0000	3.0000	4.0000	64.076
22.00	3.0000	2.0000	3.0000	3.0000	4.0000	64.975
23.00	3.0000	2.0000	3.0000	3.0000	10.000	68.385
24.00	3.0000	2.0000	3.0000	3.0000	4.0000	76.823
25.00	3.0000	2.0000	3.0000	3.0000	4.0000	83.590
26.00	3.0000	2.0000	3.0000	3.0000	4.0000	95.755
27.00	3.0000	6.0000	3.0000	3.0000	4.0000	130.55
28.00	3.0000	2.0000	1.0000	5.7575	4.0000	150.41
29.00	3.0000	2.0001	3.0000	10.000	4.0000	150.22
30.00	3.0000	2.0000	3.0000	3.0000	4.0000	164.38
31.00	3.0000	6.0000	3.0000	3.0000	8.8345	180.21
32.00	3.0000	2.0000	3.0000	3.0000	4.0000	172.68
33.00	3.0000	6.0000	3.0000	3.0000	9.9718	174.95
34.00	3.0000	2.0000	3.0000	3.0000	4.0000	135.06
35.00	3.0000	2.0000	1.0000	2.8172	4.0001	121.90
36.00	3.0000	2.0000	3.0000	3.0000	4.0000	93.817
37.00	3.0000	2.0000	3.0000	3.0000	4.0000	68.436
38.00	3.0000	2.0000	3.0000	3.0000	1.0000	53.331
39.00	3.0000	2.0000	3.0000	3.0000	4.0000	47.170
40.00	3.0000	2.0000	3.0000	3.0000	4.0000	51.095
41.00	3.0000	2.0000	3.0000	3.0000	4.0000	55.495
42.00	3.0000	2.0000	3.0000	3.0000	4.0000	62.529
43.00	3.0000	6.0000	3.0000	3.0000	4.0001	75.403
44.00	3.0000	3.8210	4.8522	1.9356	4.0000	83.426
45.00	3.0000	6.0000	4.1958	3.0000	4.0000	94.029
46.00	3.0000	6.0000	3.0000	3.0000	4.0000	106.96
47.00	3.0000	2.0000	1.0000	4.8203	4.0000	135.28
48.00	3.0000	6.0000	3.0000	3.0000	4.0000	187.03
49.00	3.0000	6.0000	3.0000	3.0000	4.2293	218.96

Lista 8.26 (continuare)

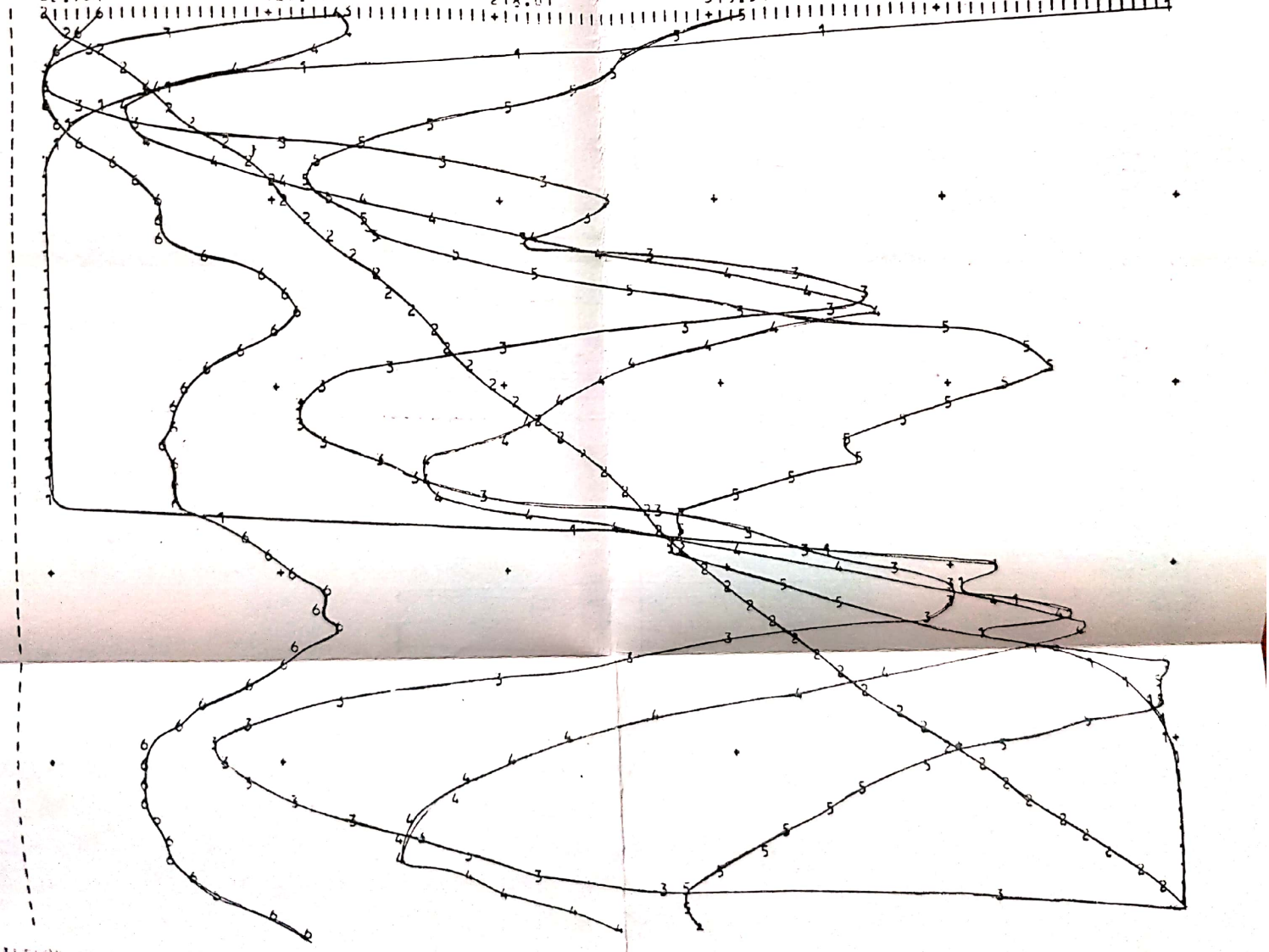
Scanned with OKEN Scanner

1	VALOAREA MAXIMA	600.08	PENTRU T	49.00	VALOAREA MINIMA	300,04	PENTRU T 26,00
2	VALOAREA MAXIMA	28762.	PENTRU T	49.00	VALOAREA MINIMA	1200.0	PENTRU T .0000
3	VALOAREA MAXIMA	436.04	PENTRU T	49.00	VALOAREA MINIMA	41.219	PENTRU T 4.000
4	VALOAREA MAXIMA	470.70	PENTRU T	34.00	VALOAREA MINIMA	56,073	PENTRU T 5.000
5	VALOAREA MAXIMA	510.61	PENTRU T	34.00	VALOAREA MINIMA	134.35	PENTRU T 9.000
6	VALOAREA MAXIMA	143.10	PENTRU T	33.00	VALOAREA MINIMA	22.951	PENTRU T 4.000

1	VALOAREA MAXIMA	3.0000	PENTRU T	.0000	VALOAREA MINIMA	3,0000	PENTRU T .0000
2	VALOAREA MAXIMA	6.0000	PENTRU T	7.000	VALOAREA MINIMA	2.0000	PENTRU T .0000
3	VALOAREA MAXIMA	4.8522	PENTRU T	44.00	VALOAREA MINIMA	1.0000	PENTRU T 28.00
4	VALOAREA MAXIMA	10.000	PENTRU T	29.00	VALOAREA MINIMA	1.0000	PENTRU T 17.00
5	VALOAREA MAXIMA	10.000	PENTRU T	23.00	VALOAREA MINIMA	1.0000	PENTRU T 38.00

V	VALOAREA MAXIMA	218.96	PENTRU T	49.00	VALOAREA MINIMA	20,308	PENTRU T 4.000

300.04	360.05	420.05	480.06	540.07	600.08
1200.0	6712.5	12225.	17737.	23250.	28762.
41.219	120.18	199.15	278.11	357.07	436.04
22.951	120.48	218.01	315.54	413.08	510.61
22.951	120.48	218.01	315.54	413.08	510.61
22.951	120.48	218.01	315.54	413.08	510.61



ILELOR DE STARE

CICLUL 1

Lista 8.27

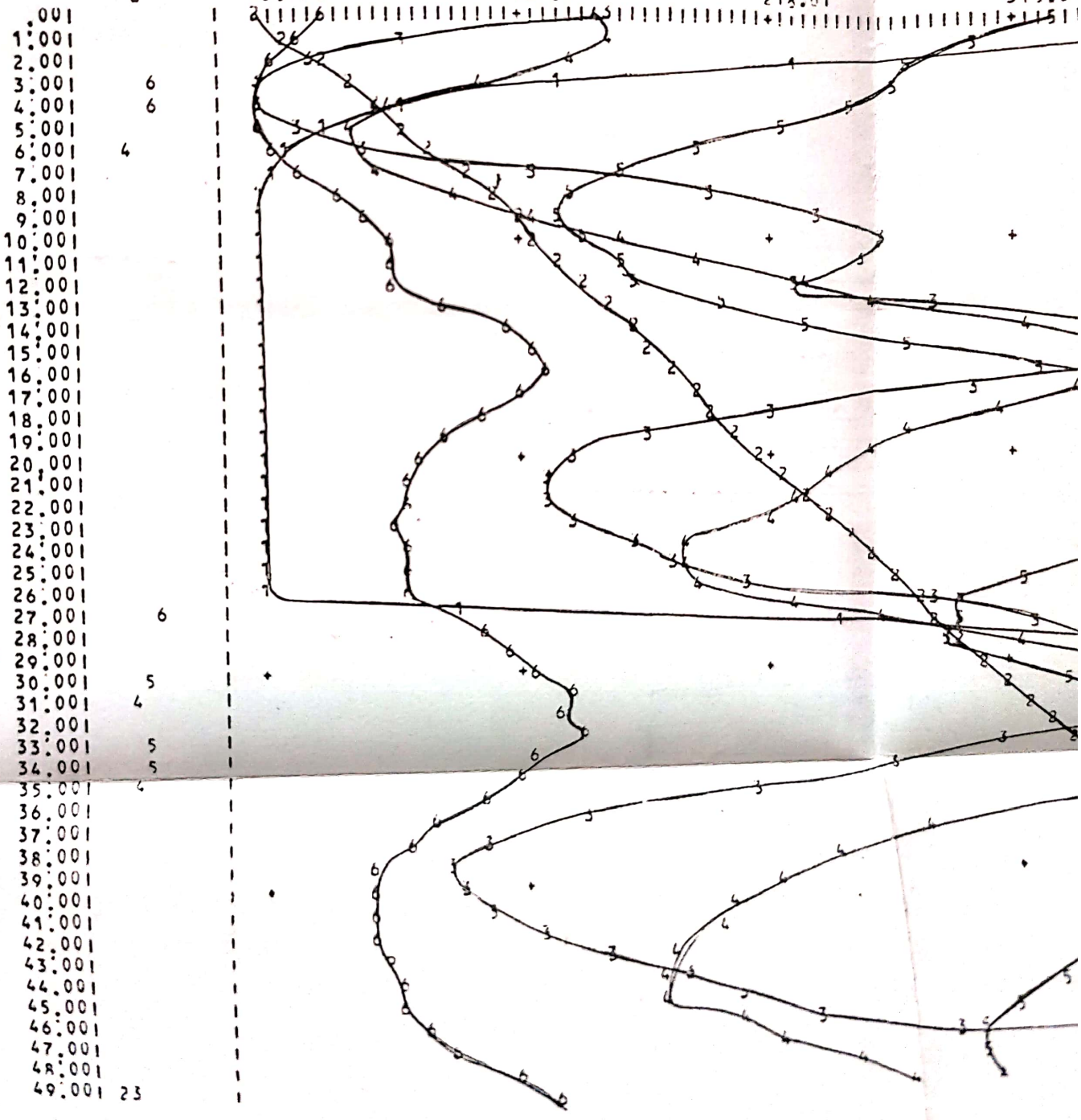
Scanned with OKEN Scanner

GRAFIC:1	X 1	VALOAREA MAXIMA	600.08	PENTRU T	49.00	VALOAR
GRAFIC:2	X 2	VALOAREA MAXIMA	28762.	PENTRU T	49.00	VALOAR
GRAFIC:3	X 3	VALOAREA MAXIMA	436.04	PENTRU T	49.00	VALOAR
GRAFIC:4	X 4	VALOAREA MAXIMA	470.70	PENTRU T	34.00	VALOAR
GRAFIC:5	X 5	VALOAREA MAXIMA	510.61	PENTRU T	36.00	VALOAR
GRAFIC:6	X 6	VALOAREA MAXIMA	143.10	PENTRU T	33.00	VALOAR

GRAFIC:1	U 1	VALOAREA MAXIMA	3.0000	PENTRU T	.0000	VALOA
GRAFIC:2	U 2	VALOAREA MAXIMA	6.0000	PENTRU T	7.000	VALOA
GRAFIC:3	U 3	VALOAREA MAXIMA	4.8522	PENTRU T	44.00	VALOA
GRAFIC:4	U 4	VALOAREA MAXIMA	10.000	PENTRU T	29.00	VALOA
GRAFIC:5	U 5	VALOAREA MAXIMA	10.000	PENTRU T	23.00	VALOA

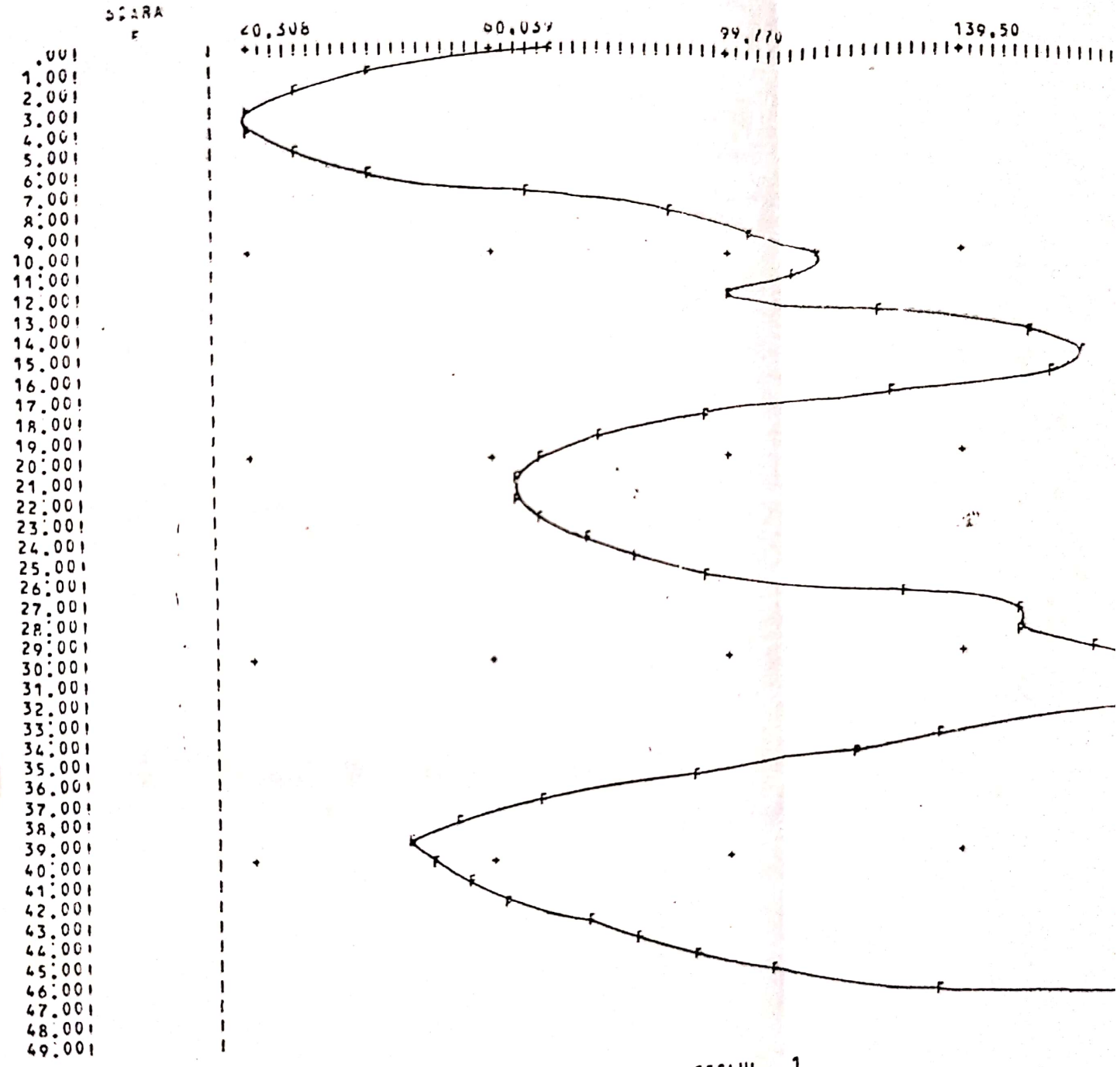
GRAFIC:F OBIECTIV VALOAREA MAXIMA 218.96 PENTRU T 49.00 VALOA

SCARA					
	1	300.04	360.05	420.05	480.06
	2	1200.0	6712.5	12225.	17737.
	3	41.219	120.18	199.15	278.11
	4	22.951	120.48	218.01	315.54
	5	22.951	120.48	218.01	315.54
	6	22.951	120.48	218.01	315.54



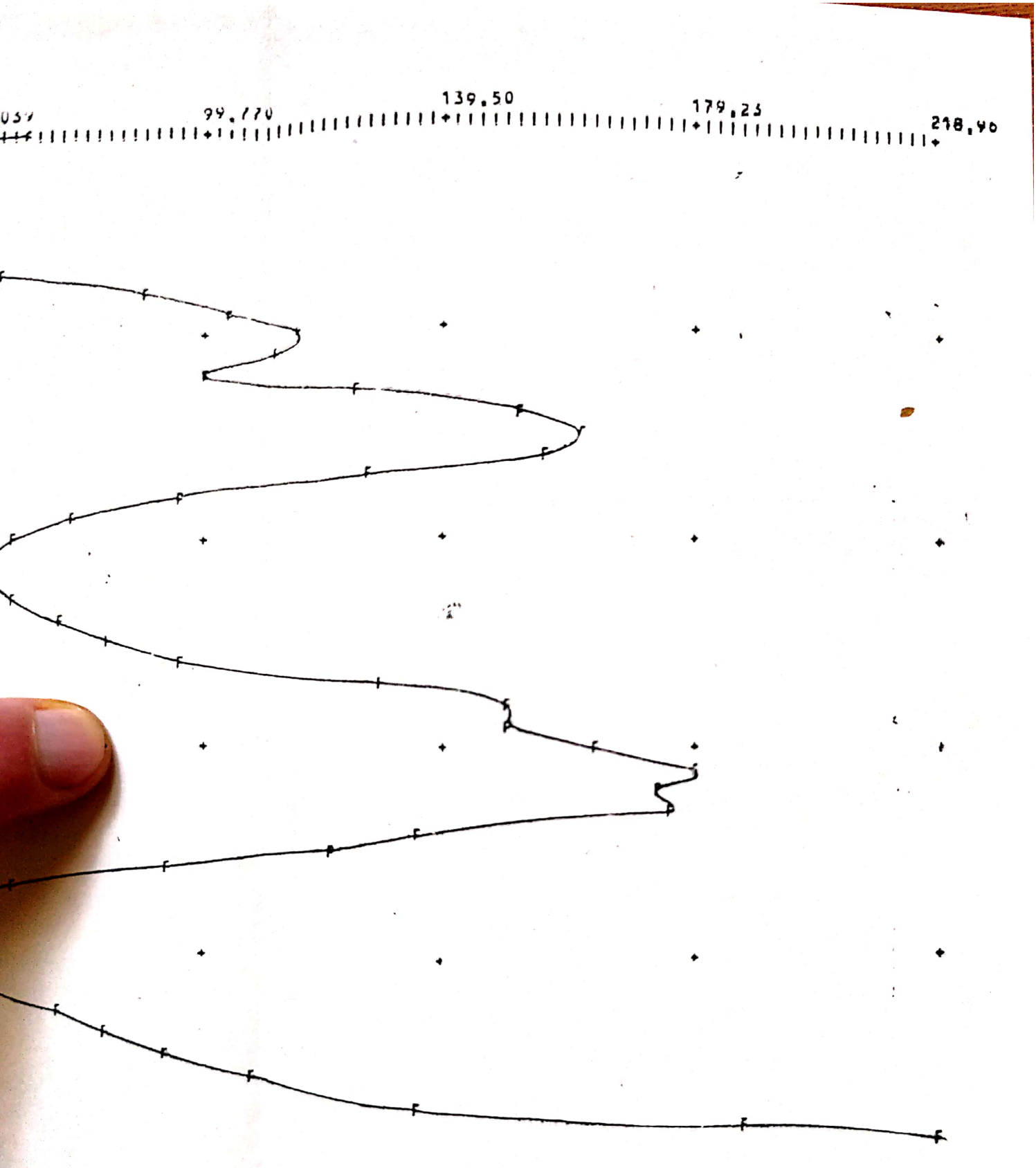
EVOLUTIA VARIABILELOR DE STARE CICLUI 1

Lista 8.27



Lista 8.27 (continuare)

Scanned with OKEN Scanner



CICLUL 1

Lista 8.27 (continuare)

Scanned with OKEN Scanner

OPTIMALITATEA SISTEMELOR
OPTIMAL V 3

VARIABILE DE STARE	N			6
VARIABILE DE COMANDA	M			5
VARIABILE DELAY	LY			1
INTERVAL DE CALCUL	DT		.100	
TIMPUL INITIAL	TI		.000	
TIMPUL FINAL	TF		50.0	
PASUL DE IMPRIMARE	PAS		1.00	
EROAREA DE CONVERGENTA SPRE OPTIM	EPS		.200	
NUMARUL MAXIM DE CICLURI	MC			3
IMPRIMAREA DATELOR INTERMEDIARE	MIT			0
MOMENTE DE TIMP CALCULATE	NIT			501
MOMENTE DE TIMP TIPARITE	NTT			50
EROAREA DE CALCUL A DERIVATIEI	ER			.100E-06
NUMERELE SIRULUI FIBONACCI	NF	6765	10946	17711
NUMARUL MAXIM DE COLOANE	NE			10
NUMARUL MAXIM DE VARIABILE	ND			50
NUMARUL MAXIM DE VARIABILE-DELAY	NY			32
NUMARUL MAXIM DE MOMENTE TIPARITE	NITD			250
NUMARUL MAXIM DE MOMENTE CALCULATE	NELIMITAT%			

COMANDA NOMINALA	VALOAREA MINIMALA	VALOAREA MAXIMALA
3.00	2.00	10.00
2.00	1.00	6.00
3.00	1.00	10.00
3.00	1.00	10.00
4.00	1.00	10.00

Lista 8.28

Programul aferent acestei scheme se găseşte catalogat în biblioteca de programe a Institutului de petrol şi gaze — Ploieşti, sub numele OPTIMAL V 3.

Comanda optimală a modelului este prezentată la sfîrşit sub forma tabelelor şi a graficelor.

Încheiem acest capitol cu rezultatele obţinute pentru comanda optimală a modelului prezentată sub formă de tabele (lista 8.26) şi de grafice (lista 8.27). În lista 8.28 sînt indicaţi parametrii programului OPTIMAL V 3 şi valorile lor pentru exemplul tratat.

Scanned with OKEN Scanner

Capitolul 9

OPTIMIZĂRI PRIN METODELE STATISTICE ALE CERCETĂRII OPERAȚIONALE

În lista cu procedee atît de diverse, intitulată „cercetare operațională", un loc aparte îl ocupă metodele de optimizare care se bazează pe aplicațiile calculului probabilității și statisticii matematice. Aceste metode sînt utilizate pentru optimizarea proceselor în care intervin variabile aleatoare.

Printre cele mai recente procedee cu caracter statistic ale cercetării operaționale se situează metodele: *teoriei firelor de așteptare*, *teoriei stocurilor* și *teoriei echipamentelor*, ale căror aplicații inginerești le vom prezenta în cadrul acestui capitol. În principiu, aceste metode se referă la menținerea stării optime (descrise prin anumite obiective) a unui sistem supus acțiunii perturbatoare a diverși factori aleatori.

Și în acest capitol, fiecare procedeu este însoțit de exemple concrete, care conțin și metodele de rezolvare prin utilizarea tehnicilor de calcul automat.

9.1. APLICAȚIILE FIRELOR DE AȘTEPTARE

Teoria firelor de așteptare a apărut relativ recent și a avut ca punct de plecare determinarea încărcării optime a unei centrale telefonice. Problema consta din urmărirea apelurilor (cererilor) care sosesc în mod întîmplător și întocmirea de statistici referitoare la timpul necesar pentru obținerea legăturilor. Se cere luarea de măsuri în așa fel încît să fie satisfăcute cît mai prompt cererile în condiții economice favorabile.

Fenomene de așteptare de acest tip apar și sînt observate în mai toate domeniile de activitate: cereri de servicii, repartizări de resurse, desfaceri de produse, deserviri, trafic etc.

Terminologie

Vom numi *client* sau *unitate* un element al mulțimii ce formează o *coadă* (*șir de așteptare*) și *stație* — punctul sau „ghișeul" în care se efectuează servicii.

Scanned with OKEN Scanner

Pentru a fi *servit* un client este necesar un anumit timp, de obicei nu același pentru toți clienții, în care stația este ocupată. Într-un asemenea caz, spunem că durata serviciului este aleatoare. De exemplu, la un ghișeu poștal un client cumpără un plic fiind servit în 30 secunde ; următorul client depune un colet ce trebuie cîntărit și pentru care se întocmesc anumite formalități, fiind servit în 2 min și 45 s etc.

Un sistem de așteptare este prezentat schematic în figura 9.1. Trebuie semnalat faptul că sistemul de așteptare prezentat în figura 9.1 este un sistem cu un singur șir de așteptare și o singură stație de servire. Există și alte modalități în care sînt organizate sistemele de așteptare, cum ar fi sisteme cu două sau mai multe stații de servire paralele (fig. 9.2) și în serie (fig. 9.3), cu unul sau mai multe șiruri de așteptare etc.

Să considerăm, în continuare, cazul din figura 9.1 în care există un singur șir de așteptare și o singură stație de servire. Fiecare client (unitate) intră în incinta unde funcționează stația de servire și preia ultimul loc din

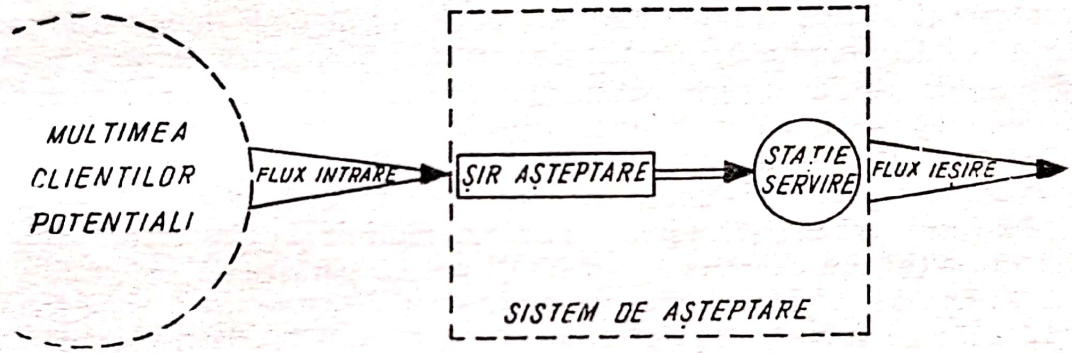


Fig. 9.1

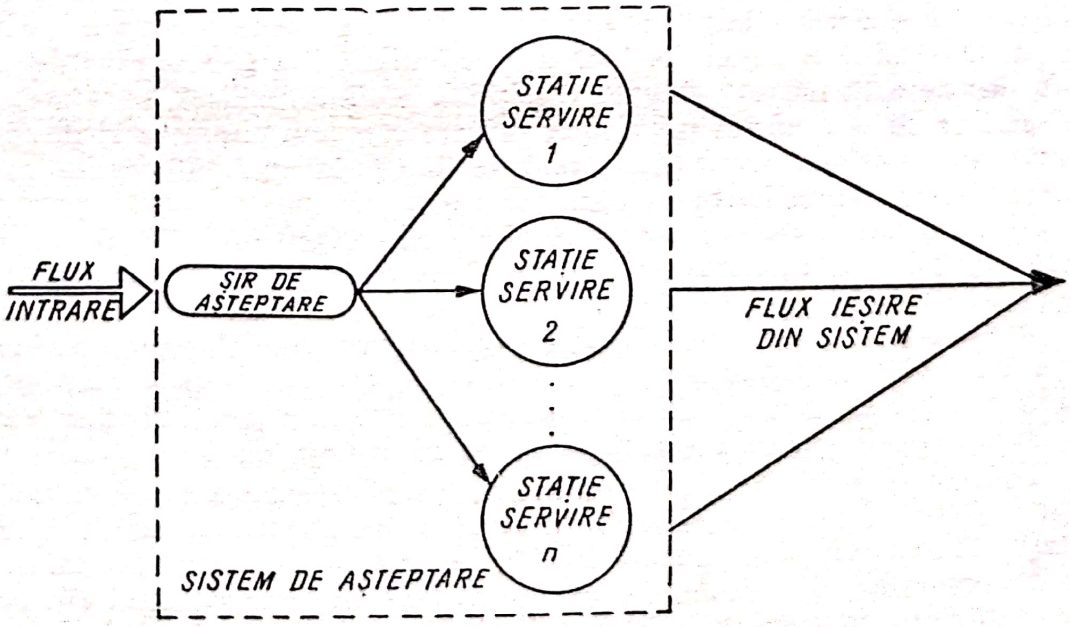


Fig. 9.2

Scanned with OKEN Scanner

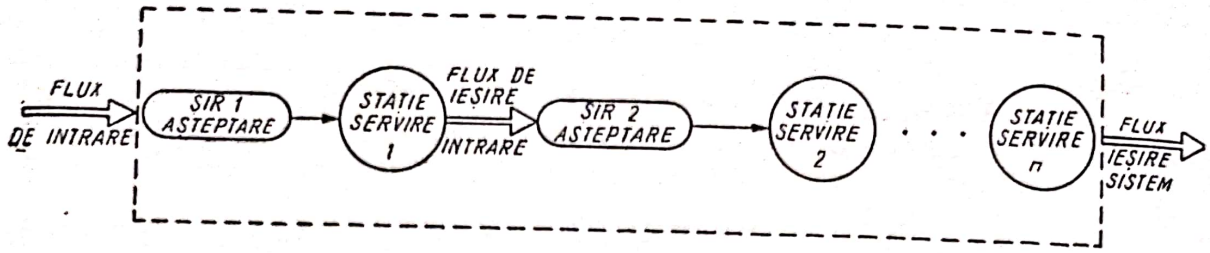


Fig. 9.3

coadă. În momentul în care un client este servit, acesta pleacă și un altul din șirul de așteptare îi ia locul după o anumită regulă, numită *disciplina șirului de așteptare*. Cele expuse mai sus ne sînt utile pentru completarea vocabularului utilizat în teoria șirurilor de așteptare.

Fluxul de intrare caracterizează modul cum intră în sistem clienții. În cele mai multe cazuri, se presupune (și apoi se verifică prin diferite teste) că intrările în sistem sînt aleatoare și independente. Prin aceasta înțelegem că o intrare în sistem în intervalul $(t,\ t + dt)$ nu depinde de intrările precedente și nici de numărul de clienți existenți în sistem. Dacă, în plus, intrările nu depind de faptul că observațiile sînt făcute la anume ore (de exemplu, 8 sau 19), se spune că sistemul este în *regim staționar*. Probabilitatea ca în intervalul de timp $(t,\ t + dt)$ să se producă o intrare este egală cu $1/\lambda$, unde λ este media intrărilor în sistem în decursul unui interval de timp dt. Numărul clienților care intră în sistem poate fi finit sau infinit și de asemenea, dintr-un motiv sau altul, un client sau mai mulți pot să părăsească sistemul fără a fi serviți. În acest ultim caz spunem că sistemul de așteptare este *cu pierderi*.

În ceea ce privește modul cum sosesc clienții în sistem, se constată (de la caz la caz) că sosirile pot fi *individuale* sau *în grup*.

Se numește *șir (fir) de așteptare* mulțimea clienților care așteaptă să fie serviți, mai puțin cel care este în curs de servire. Este de remarcat faptul că fiecare șir de așteptare e dotat cu o *disciplină de așteptare*, disciplină care precizează modul de formare a șirului și modul cum vor fi serviți clienții. Disciplina cea mai des întîlnită este disciplina *primul venit—primul servit*, dar aceasta nu este singura disciplină posibilă. Există, spre exemplu, și discipline cu o anumită prioritate, printre care am enumera și disciplina *ultimul sosit—primul servit* (este cazul stațiilor radar unde ultima informație detectată este prima luată în considerare).

Se numește *sistem de așteptare* mulțimea clienților care formează o coadă, fiind inclus în această mulțime și cel care este servit în momentul respectiv.

Am arătat că un sistem de așteptare poate avea una, două sau mai multe stații de servire. Numărul lor se va nota cu S. Din cele expuse mai înainte, rezultă imediat că $S \in N$. În cazul în care S este infinit toți clienții ce intră în sistem vor fi imediat serviți, neexistînd așteptare.

În ceea ce privește serviciul, acesta poate fi executat individual sau în grup (bloc). În cazul *serviciului individual* un client este servit de către o stație. Cînd serviciul se execută *în bloc*, o stație servește simultan un număr (limitat superior) de clienți. Ca exemplu dăm traversarea unui rîu cu un bac. Autovehiculele sosesc la locul de îmbarcare și formează un șir de așteptare în timp ce se așteaptă sosirea bacului. La sosirea bacului, un număr

Scanned with OKEN Scanner

finit de autovehicule (atîtea cîte încap pe bac) este îmbarcat — deci începe servirea, iar restul aşteaptă bacul următor.

Timpul necesar servirii unui client (grup) se numeşte *timp de servire.*

Fluxul de ieşire va fi luat în studiu de obicei în sistemele de aşteptare cu cele S staţii aşezate în serie (v. fig. 9.3), deoarece fluxul de ieşire de la staţia i devine flux de intrare pentru staţia $i + 1 (i = 1, 2, \ldots, n - 1$, ceea ce se mai scrie şi $i = \overline{1, n-1}$).

Timpul de aşteptare al unui client este timpul cît clientul stă în şirul de aşteptare înainte de a fi servit.

Timpul total de rămînere în sistem este egal cu suma timpilor de aşteptare şi de servire.

Notînd cu λ numărul mediu de intrări în sistem în unitate de timp şi cu μ numărul mediu de serviri în unitate de timp se defineşte *factorul de serviciu* ρ ca fiind raportul:

$$\rho = \frac{\lambda}{\mu}. \tag{9—1}$$

Factorul de serviciu are o importanţă deosebită, deoarece după determinarea repartiţiei timpului de servire, toate caracteristicile modelului ce va fi stabilit se vor exprima în funcţie de acest parametru; deci evoluţia sistemului depinde numai de λ, μ şi deci de ρ.

Într-o problemă de fire de aşteptare se cere, în general, să se determine numărul staţiilor de servire în aşa fel încît clienţii să nu piardă decît un timp „rezonabil". Rezolvarea logică a problemei ar fi mărirea numărului de staţii de servire, dar aceasta ar duce la creşterea costului serviciului prin cheltuirea de sume importante pentru noi utilaje şi creşterea costului de întreţinere. Problema revine la a minimiza costul total de întreţinere a staţiilor şi de aşteptare a clienţilor, deci în a stabili un „compromis" între *preţul de servire* şi *preţul de aşteptare* al clienţilor, aceste două preţuri fiind invers proporţionale. Preţul de servire este direct proporţional cu numărul staţiilor, în timp ce acela de aşteptare al clienţilor depinde de date aleatoare, cum sînt fluxul de intrare şi durata serviciilor.

Orice sistem de aşteptare poate fi descris de un model matematic în funcţie de fluxul de intrare, repartiţia timpului de servire şi numărul staţiilor de servire.

Clasificarea modelelor de aşteptare

Avînd în vedere fluxul de intrare, repartiţia timpului de servire şi numărul de staţii, se poate obţine o clasificare a modelelor de aşteptare, aşa cum este clasificarea făcută de Kendal.

În funcţie de fluxul de intrare se pot prezenta situaţiile:

1° Clienţii sosesc la intervale egale de timp t, cu $0 < t < \infty$; fluxul se va numi *determinat* (D). În acest caz, funcţia de repartiţie a numărului de clienţi ce intră în sistem este:

$$F(x) = \begin{cases} 1 \text{ pentru } x \geqslant t \\ 0 \text{ pentru } x < t \end{cases} \tag{9—2}$$

2° Dacă clienţii sosesc în sistem după un proces Poisson de parametru λ, cu $0 < \lambda < \infty$, λ fiind intensitatea fluxului, fluxul se numeşte *aleator*

Scanned with OKEN Scanner

sau *poissonian* (P). Numărul clienților care intră în sistem în intervalul $(0, t]$ are repartiția Poisson $\frac{(\lambda t)^n e^{-\lambda t}}{n!}$. Se demonstrează că repartiția intervalelor dintre două intrări consecutive urmează o lege exponențială și anume:

$$F(t) = \lambda e^{-\lambda t}. \tag{9—3}$$

3° Dacă funcția de repartiție a numărului clienților ce intră în sistem este dată de:

$$F(t) = \frac{(k/a)^k}{(k-1)!} t^{k-1} e^{-\frac{kt}{a}}, \tag{9—4}$$

cu $0 < t < \infty$, $k \in N$ și a reprezentînd media lungimii intervalelor dintre două intrări succesive în sistem, fluxul de intrare se numește *Erlang* (E_k). Se poate observa că fluxul de intrare Erlang pentru $k = 1$ se transformă în flux Poisson.

4° Dacă asupra funcției de repartiție F a numărului clienților ce intră în sistem nu se face nici o ipoteză în afara existenței valorii medii finite a $(0 < a < \infty)$, fluxul de intrare se numește *general independent* (GI).

Avînd în vedere timpul de servire, respectiv funcția de repartiție $H(t)$ a timpului de servire, cele expuse mai sus despre fluxul de intrare, respectiv funcția de repartiție $F(t)$ a numărului de clienți ce intră în sistem, rămîn valabile și pentru $H(t)$. Deci, la rîndul său, funcția de repartiție $H(t)$ poate fi: determinată (D), poissoniană (P), de tip Erlang (E_k) și, în sfîrșit, dacă asupra repartiției $H(t)$ a timpului de servire nu se poate face nici o presupunere decît doar existența valorii medii a $(0 < a < \infty)$, general (G). Ținînd cont de cele expuse mai sus și notînd cu S numărul stațiilor de servire din sistem ($S \in N$), clasele de modele de așteptare se pot reprezenta sub forma unei tabele (așa ca în tabela 9—1).

Tabela 9—1

Parametrii de clasificare	Clase de modele de așteptare															
$F(x)$	D	D	D	D	P	P	P	P	E_k	E_k	E_k	E_k	GI	GI	GI	GI
$H(t)$	D	P	E_k	G	D	P	E_k	G	D	P	E_k	G	D	P	E_k	G
S	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*	S^*

În tabela de mai sus, S^* indică faptul că fiecare clasă se descompune, la rîndul ei, în subclase și anume: pentru $S = 1$, pentru $1 < S < \infty$ și pentru $S \to \infty$. Așa de exemplu, clasa $DDS^* = DD1 \cup DDS \cup DD\infty$ unde $1 \leqslant S < \infty$ și $S \in N$.

În continuare, vom lua în studiu clasa PPS^*, deoarece în multe sisteme de așteptare legile sosirilor și a servirilor sînt Poisson și respectiv exponențiale.

Scanned with OKEN Scanner

Modele de aşteptare ale clasei PPS*

Am arătat că studiul fenomenelor de aşteptare are ca scop final optimizarea sistemului prin criterii economice. De asemenea, am mai arătat că în cazul clasei PPS^* repartiţia numărului de clienţi în sistemul de aşteptare se face după o distribuţie Poisson. Ţinînd seama de aceste două elemente fundamentale ca şi de celelalte caracteristici al clasei PPS^*, vom presupune următoarele :

1) sosirile şi servirile sînt poissoniene ;

2) fenomenul de aşteptare este staţionar (caracteristicile sale nu depind de originea timpului de observaţie) ;

3) sistemul este cu o singură staţie, parametrul λ reprezintă numărul mediu de sosiri în unitatea de timp, iar μ este numărul mediu de serviri (mai jos, vom arăta cum pot fi reprezentaţi parametrii sistemului de aşteptare în funcţie de λ, μ şi factorul de serviciu ρ) ;

4) vom considera disciplina primul sosit—primul servit.

Vom folosi, în continuare, următoarele notaţii :

$P_n(t)$ — probabilitatea de a exista n clienţi în sistem la momentul t ;
$P_0(t)$ — probabilitatea de a nu fi aşteptare la momentul t ;
$P(t)$ — probabilitatea de a fi aşteptare la momentul t ;
$\overline{n}_s$ — numărul mediu de clienţi în sistem ;
$\overline{n}_f$ — numărul mediu de clienţi în şirul (firul) de aşteptare ;
$\overline{t}_s$ — timpul mediu de aşteptare în sistem ;
$\overline{t}_f$ — timpul mediu de aşteptare în şir (fir) ;
$\overline{\rho}$ — rata medie de inactivitate a staţiilor din sistem.

Să luăm acum în studiu fluxul de intrare a unui sistem de aşteptare ce satisface condiţiile cerute mai sus. Presupunem că la momentul $t + \Delta t$ au fost observate n sosiri. Dacă se ia în considerare şi momentul precedent, t, situaţia de a avea la momentul $t + \Delta t$ n sosiri poate proveni din unul din următoarele două cazuri :

1° sau la momentul t au fost observate $n - 1$ sosiri şi la $t + \Delta t$ n sosiri ;

2° sau la momentul t au fost observate n sosiri şi la $t + \Delta t$ tot n sosiri. Atunci :

$$P_n(t + \Delta t) = P_{n-1}(t) \cdot \lambda \cdot \Delta t + P_n(t)(1 - \lambda \cdot \Delta t) \qquad (9—5)$$

aceasta deoarece $\lambda \cdot \Delta t$ este probabilitatea de a se observa o sosire în intervalul de timp Δt, iar $1 - \lambda \cdot \Delta t$ este probabilitatea complementară şi indiferent dacă sîntem în cazul 1° la $t + \Delta t$ vom avea observate în mod sigur n sosiri.

Mai observăm că relaţia (9—5) este valabilă pentru $\forall n \in N$ şi $n \neq 0$, deoarece dacă nu a fost observată nici o sosire la momentul $t + \Delta t$ rezultă că nici la momentul t nu există posibilitatea să fie observată vreo sosire. Deci în cazul în care $n = 0$ relaţia (9—5) devine :

$$P_0(t + \Delta t) = P_0(t)(1 - \lambda \cdot \Delta t), \qquad (9—6)$$

Scanned with OKEN Scanner

de unde rezultă:

$$\frac{P_0(t+\Delta t)-P_0(t)}{\Delta t}=-\lambda P_0(t),$$

sau

$$\lim_{\Delta t\to 0}\frac{P_0(t+\Delta t)-P_0(t)}{\Delta t}=-\lambda P_0(t).$$

Avînd în vedere definiţia derivatei unei funcţii într-un punct şi presupunînd că $\lim_{\Delta t\to 0}\frac{P_0(t+\Delta t)-P_0(t)}{\Delta t}$ există şi este finită, atunci relaţia de mai sus se poate scrie:

$$P_0'(t)=-\lambda P_0(t)$$

sau încă:

$$\frac{dP_0(t)}{dt}=-\lambda P_0(t)\Rightarrow\frac{dP_0(t)}{P_0(t)}=-\lambda\,dt. \tag{9—7}$$

Integrînd ecuaţia diferenţială (9—7) obţinem:

$$\ln P_0(t)=-\lambda t+\ln C\Rightarrow\ln\frac{P_0(t)}{C}=-\lambda t\Rightarrow P_0(t)=Ce^{-\lambda t}. \tag{9—8}$$

Pentru determinarea lui C vom ţine cont de faptul că la momentul $t=0$, $P_0(0)=1$, deoarece la $t=0$ faptul că nu avem nici o sosire este cert. Din (9—8) şi $P_0(0)=1$ rezultă:

$$Ce^{-\lambda\cdot 0}=1\Rightarrow C=1.$$

Deci soluţia ecuaţiei diferenţiale (9—7) este:

$$P_0(t)=e^{-\lambda t}. \tag{9—9}$$

În mod asemănător din relaţia (9—5) pentru $n\neq 0$ deducem succesiv:

$$P_n(t+\Delta t)-P_n(t)=(P_{n-1}(t)-P_n(t))\lambda\cdot\Delta t,$$

$$\frac{P_n(t+\Delta t)-P_n(t)}{\Delta t}=-\lambda P_n(t)+\lambda P_{n-1}(t), \tag{9—10}$$

$$\lim_{\Delta t\to 0}\frac{P_n(t+\Delta t)-P_n(t)}{\Delta t}=-\lambda P_n(t)+\lambda P_{n-1}(t),$$

$$P_n'(t)=-\lambda P_n(t)+\lambda P_{n-1}(t).$$

Făcînd pe rînd $n=1, 2, \ldots$ în relaţia (9—10) vom obţine succesiv soluţiile prezentate mai jos:

Pentru $\boxed{n=1}$

$$P_1(t)=-\lambda P_1(t)+\lambda P_0(t)$$

şi folosind relaţia (9—9) deducem:

$$P_1'(t)=-\lambda P_1(t)+\lambda e^{-\lambda t},$$

adică:

$$P_1'(t) + \lambda P_1(t) - \lambda e^{-\lambda t} = 0. \qquad (9-11)$$

Expresia (9—11) este o ecuație diferențială liniară de ordinul întîi de forma:

$$y'(t) + P(t)y(t) + Q(t) = 0,$$

care are soluția generală dată de:

$$y(t) = e^{-\int P(t)dt}(c - \int Q(t)e^{\int P(t)dt}dt).$$

În cazul ecuației (9—11) soluția generală va fi:

$$P_1(t) = e^{-\int \lambda dt}(c + \int \lambda e^{-\lambda t}e^{\int \lambda dt}) = e^{-\lambda t}(c + \lambda t) \qquad (9-12)$$

Se pune acum problema determinării constantei de integrare c din (9—12). Pentru aceasta, observăm că este imposibil ca la momentul $t = 0$ să avem o sosire în sistem. Deci:

$$P_1(0) = 0 \Rightarrow P_1(0) = e^{-\lambda \cdot 0}c + \lambda \cdot 0 \Rightarrow c = 0,$$

astfel încît soluția ecuației (9—11), de forma (9—12), devine:

$$P_1(t) = \lambda t e^{-\lambda t}. \qquad (9-13)$$

Pentru $\boxed{n = 2}$

$$P_2'(t) + \lambda P_2(t) - \lambda P_1(t) = 0, \qquad (9-14)$$

care se obține din (9—10) și are soluția generală de forma:

$$P_2(t) = e^{-\lambda t}\left(c + \frac{\lambda^2 t^2}{2}\right).$$

Determinarea constantei c se face în același mod ca mai sus, găsindu-se $c = 0$, de unde rezultă soluția ecuației (9—14) și anume:

$$P_2(t) = \frac{\lambda^2 t^2}{2} e^{-\lambda t}. \qquad (9-15)$$

Pentru $\boxed{n = 3}$ va rezulta similar:

$$P_3(t) = \frac{\lambda^3 t^3}{6} e^{-\lambda t}. \qquad (9-16)$$

Observăm că relațiile (9—13), (9—15) și (9—16) se mai pot scrie și sub forma:

$$P_1(t) = \frac{\lambda t}{1!} e^{-\lambda t}, \qquad (9-13, a)$$

$$P_2(t) = \frac{(\lambda t)^2}{2!} e^{-\lambda t}, \qquad (9-15, a)$$

$$P_3(t) = \frac{(\lambda t)^3}{3!} e^{-\lambda t}. \qquad (9-16, a)$$

Scanned with OKEN Scanner

Prin recurență putem scrie:

$$P_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}. \qquad (9-17)$$

Valabilitatea relației (9—17) o vom demonstra folosind metoda inducției matematice.

Etapa I a metodei inductive a fost verificată pentru $n = 1, 2, 3$ prin găsirea relațiilor (9—13), (9—15) și (9—16).

Etapa a II-a. Vom presupune că relația (9—17) este valabilă pentru un $n \in N$ și să demonstrăm că este valabilă și pentru succesorul său, $n + 1$. Deci în relația (9—11) $n := n + 1$* rezultînd:

$$P_{n+1}(t) + \lambda P_{n+1}(t) - \lambda P_n(t) = 0,$$

sau ținînd cont de ipoteza inductivă obținem ecuația diferențială:

$$P_{n+1}(t) + \lambda P_{n+1}(t) - \frac{\lambda^{n+1} t^n}{n!} e^{-\lambda t} = 0,$$

cu soluția generală:

$$P_{n+1}(t) = e^{-\lambda t}\left[c + \frac{\lambda^{n+1} t^{n+1}}{(n+1)!}\right].$$

Cum la momentul $t = 0$ este imposibil să avem $n + 1$ sosiri, rezultă că $P_{n+1}(0) = 0$ și de aici $c = 0$.

Deci $P_{n+1}(t) = \frac{(\lambda t)^{n+1}}{(n+1)!} e^{-\lambda t}$ și deci relația (9—17) este valabilă pentru $\forall n \in N$.

Valabilitatea relației (9—17) demonstrează că într-un fenomen de așteptare, satisfăcînd ipotezele făcute, legea de distribuție a numărului de sosiri într-un interval de timp t, este poissoniană cu parametrul λt.

Să determinăm acum și repartiția timpului dintre două sosiri consecutive în sistemul de așteptare.

Fie X variabila aleatoare reprezentînd timpul dintre două sosiri consecutive. Să luăm ca moment inițial momentul în care a sosit primul client. Probabilitatea ca intervalul de timp X dintre două sosiri consecutive să fie mai mare decît un interval de timp t este:

$$P(X > t) = P_0(t) = e^{-\lambda t},$$

deoarece în intervalul de timp t trebuie să nu se producă nici o sosire. Atunci probabilitatea complementară va fi:

$$P(X \leqslant t) = 1 - e^{-\lambda t},$$

care reprezintă funcția de repartiție a variabilei aleatoare X. Deci:

$$F(t) = 1 - e^{-\lambda t},$$

---

* Prin $n := n + 1$ se înțelege faptul că „se atribuie" lui n valoarea $n + 1$.

Scanned with OKEN Scanner

iar densitatea de probabilitate este dată de:

$$f(t) = F'(t) = \lambda e^{-\lambda t}, \text{ cu } t \geqslant 0.$$

Deci repartiția intervalelor de timp dintre două sosiri consecutive este o repartiție exponențială.

Media acestei repartiții va fi:

$$M(x) = \int_0^{\infty} t \cdot f(t)\, dt = \int_0^{\infty} t\lambda e^{-\lambda t}\, dt = \frac{1}{\lambda}.$$

Să observăm că intervalul mediu dintre două sosiri consecutive este $1/\lambda$, deci inversul numărului mediu de sosiri în unitate de timp.

S-a constatat, că o asemenea lege este urmată adeseori de duratele de servire. Dacă μ este numărul de clienți serviți în unitate de timp, atunci $1/\mu$ va fi timpul mediu consumat de fiecare client la stația de servire.

Să luăm acum în considerare sistemul de așteptare în ansamblu, adică să ținem cont atît de fluxul de intrare cît și de fluxul de ieșire.

Presupunem că în sistemul de așteptare la momentul $t + \Delta t$ sînt n clienți. Această situație poate apare în unul din următoarele patru cazuri:

a) la momentul t sînt în sistem $n - 1$ clienți și în intervalul Δt intră un client și nu părăsește sistemul nici unul,

b) la momentul t sînt în sistem n clienți și în intervalul Δt nu intră și nici nu părăsește sistemul nici un client,

c) la momentul t sînt în sistem n clienți și în intervalul Δt intră și părăsește sistemul cîte un client,

d) la momentul t sînt în sistem $n + 1$ clienți, nu intră nici unul, dar părăsește sistemul un singur client.

Cum probabilitatea de intrare a unui client în sistem în intervalul Δt este $\lambda \Delta t$ și cea de ieșire este $\mu \Delta t$, iar probabilitățile complementare (de a nu intra și a nu ieși din sistem) sînt $1 - \lambda \Delta t$ și $1 - \mu \Delta t$, putem scrie relația:

$$P_n(t + \Delta t) = P_{n-1}(t) \cdot \lambda \cdot \Delta t(1 - \mu \Delta t) + P_n(t)(1 - \lambda \Delta t)(1 - \mu \Delta t) +$$
$$+ P_n(t) \cdot \lambda \cdot \Delta t \cdot \mu \cdot \Delta t + P_{n+1}(t)(1 - \lambda \Delta t) \cdot \mu \Delta t.$$

Dar cum Δt este practic suficient de mic atunci $\Delta t^2 \to 0$ și relația de mai sus devine:

$$P_n(t + \Delta t) = P_{n-1}(t) \cdot \lambda \Delta t + P_n(t)(1 - (\lambda + \mu) + P_{n+1}(t)\mu \Delta t,$$

sau:

$$\frac{P_n(t + \Delta t) - P_n(t)}{\Delta t} = \lambda P_{n-1}(t) - (\lambda + \mu) P_n(t) + \mu P_{n+1}(t), \quad (9-18)$$

relație valabilă pentru $\forall n \in N$.

În cazul cînd $n = 0$, deci la momentul $t + \Delta t$, avem în sistemul de așteptare 0 clienți; cele patru cazuri enumerate mai sus se reduc la trei și anume:

a') la momentul t sînt 0 clienți în sistem, nu intră și nu părăsește sistemul nici unul în intervalul de timp Δt;

Scanned with OKEN Scanner

b') la momentul t sînt 0 clienţi în sistem, intră unul şi părăseşte sistemul unul în intervalul Δt;

c') la momentul t există în sistem un client, nu intră nici un client dar părăseşte sistemul un singur client în intervalul de timp Δt.

Avînd aceste situaţii (pentru $n = 0$), vom observa că dacă în sistem sînt 0 clienţi la t, probabilitatea ca el să nu iasă din sistem în intervalul de timp Δt este 1 şi nu $1 - \mu\Delta t$. Ţinînd cont de acestea, pentru $n = 0$ relaţia (9—18) devine:

$$\frac{P_0(t + \Delta t) - P_0(t)}{\Delta t} = \lambda P_0(t) + \mu P_1(t). \qquad (9—18,\ a)$$

Trecînd la limită pentru $\Delta t \to 0$ în (9—18) şi (9—18, *a*) obţinem:

$$P_n'(t) = \lambda P_{n-1}(t) - (\lambda + \mu)P_n(t) + \mu P_{n+1}(t), \qquad (9—19)$$

$$P_0'(t) = -\lambda P_0(t) + \mu P_1(t). \qquad (9—19,\ a)$$

Dar cum fenomenul este staţionar rezultă:

$$P_0'(t) = P_1'(t) = \ldots = P_n'(t) = 0$$

şi ecuaţiile diferenţiale (9—19) şi (9—19, *a*) se transformă în ecuaţii liniare:

$$\mu P_1(t) - \lambda P_0(t) = 0,$$

adică:

$$P_1(t) = \frac{\lambda}{\mu} P_0(t).$$

Pentru $n := 1$ în relaţia (9—19) se obţine:

$$\lambda P_0(t) - (\lambda + \mu)P_1(t) + \mu P_2(t) = 0,$$

$$P_2(t) = \left(\frac{\lambda}{\mu}\right)^2 P_0(t).$$

În general, pentru $n := n$ avem:

$$P_n(t) = \left(\frac{\lambda}{\mu}\right)^n P_0(t), \qquad (9—20)$$

relaţia ce se demonstrează prin inducţie matematică completă.

Pentru a determina $P_0(t)$ vom porni de la relaţia:

$$P_0(t) + P_1(t) + \ldots + P_n(t) + \ldots = 1$$

şi folosind relaţia (9—20) obţinem:

$$P_0(t) + P_1(t) + \ldots + P_n(t) + \ldots = P_0(t) + \frac{\lambda}{\mu} P_0(t) +$$

$$+ \left(\frac{\lambda}{\mu}\right)^2 P_0(t) + \ldots + \left(\frac{\lambda}{\mu}\right)^n P_0(t) + \ldots = P_0(t)\left[1 + \frac{\lambda}{\mu} + \left(\frac{\lambda}{\mu}\right)^2 + \ldots +\right.$$

$$\left. + \left(\frac{\lambda}{\mu}\right)^n + \ldots\right] = P_0(t) \frac{1 - \left(\frac{\lambda}{\mu}\right)^n}{1 - \frac{\lambda}{\mu}}.$$

Scanned with OKEN Scanner

Vom observa că se impune: $\frac{\lambda}{\mu} < 1$, ca suma din paranteză să aibă sens (ca interpretare, pentru ca firul de aşteptare să nu crească nemărginit).

Cum $\frac{\lambda}{\mu} < 1 \Rightarrow \left(\frac{\lambda}{\mu}\right)^n \to 0$ şi pentru n suficient de mare putem face aproximarea:

$$\sum_{i \in N \cup \{0\}} P_i(t) = P_0(t) \frac{1}{1 - \frac{\lambda}{\mu}}.$$

şi deoarece: $\sum_{i \in N \cup \{0\}} P_i(t) = 1$, rezultă:

$$P_0(t) = 1 - \frac{\lambda}{\mu} = 1 - \rho, \qquad (9-21)$$

astfel încît relaţia (9—20) devine:

$$P_n(t) = \left(\frac{\lambda}{\mu}\right)^n \left(1 - \frac{\lambda}{\mu}\right) = \rho^n(1 - \rho). \qquad (9-20,a)$$

Media acestei distribuţii fiind:

$$\bar{n}_s = \sum_{i \in N \cup \{0\}} iP_i(t) = \left[\frac{\lambda}{\mu} + 2\left(\frac{\lambda}{\mu}\right)^2 + \ldots + n\left(\frac{\lambda}{\mu}\right)^n + \ldots\right] \times$$

$$\times \left(1 - \frac{\lambda}{\mu}\right) = \left(1 - \frac{\lambda}{\mu}\right) \frac{\lambda}{\mu} \left[1 + 2\frac{\lambda}{\mu} + 3\left(\frac{\lambda}{\mu}\right)^2 + \ldots + n\left(\frac{\lambda}{\mu}\right)^{n-1} + \ldots\right].$$

Dar în paranteza dreaptă avem derivata în raport cu $\frac{\lambda}{\mu}$ a progresiei:

$$S = \frac{\lambda}{\mu} + \left(\frac{\lambda}{\mu}\right)^2 + \left(\frac{\lambda}{\mu}\right)^3 + \ldots + \left(\frac{\lambda}{\mu}\right)^n = \frac{1}{1 - \frac{\lambda}{\mu}},$$

adică:

$$S = \frac{\lambda}{\mu}\left(\frac{1}{\left(1 - \frac{\lambda}{\mu}\right)^2}\right)$$

şi deci:

$$\bar{n}_s = \left(1 - \frac{\lambda}{\mu}\right) \cdot \frac{\lambda}{\mu} \cdot \frac{1}{\left(1 - \frac{\lambda}{\mu}\right)^2} = \frac{\frac{\lambda}{\mu}}{1 - \frac{\lambda}{\mu}},$$

sau:

$$\bar{n}_s = \frac{\rho}{1 - \rho} = \frac{\lambda}{\mu - \lambda}. \qquad (9-22)$$

Scanned with OKEN Scanner

Timpul mediu de aşteptare în sistem va fi atunci:

$$\bar{t}_s = \frac{\bar{n}_s}{\lambda} = \frac{1}{\lambda} \cdot \frac{\rho}{1-\rho} = \frac{1}{\mu} \cdot \frac{1}{1-\rho}, \tag{9—23}$$

dar cum $\bar{t}_s - \bar{t}_f = \frac{1}{\mu}$ rezultă:

$$\bar{t}_f = \frac{\rho}{\mu(1-\rho)} = \frac{1}{\mu - \lambda} - \frac{1}{\lambda}. \tag{9—24}$$

Rata medie de inactivitate va fi atunci:

$$\bar{\rho} = S - \rho, \tag{9—25}$$

iar probabilitatea de a exista aşteptare este deci:

$$P(t) = \rho, \tag{9—26}$$

ceea ce rezultă din faptul că probabilitatea de a nu exista aşteptare este $P_0(t) = 1 - \rho$ şi cum $P(t)$ este complementară se obţine imediat (9—26).

În sfîrşit, să calculăm $\bar{n}_f$. Deoarece sîntem în cazul unui sistem cu o singură staţie şi dacă avem în sistem n clienţi atunci:

$$\bar{n}_f = n - 1$$

şi

$$\bar{n}_f = \sum_{n_f=1}^{\infty} n_f P_{n_f}(t) = \sum_{n=2}^{\infty} (n-1) P_n(t) = \sum_{n=2}^{\infty} n P_n(t) - \sum_{n=2}^{\infty} P_n(t) =$$

$$= \sum_{n=0}^{\infty} n P_n(t) - \sum_{n=0}^{1} n P_n(t) - \sum_{n=0}^{\infty} P_n(t) + \sum_{n=0}^{1} P_n(t) =$$

$$= \bar{n}_s - P_1(t) - 1 + P_0(t) + P_1(t) = \bar{n}_s - 1 + P_0(t).$$

Folosind relaţiile (9—21) şi (9—20) obţinem:

$$\bar{n}_f = \frac{\rho}{1-\rho} - \rho = \frac{\rho^2}{1-\rho}. \tag{9—27}$$

Cu aceasta, modelul (matematic) al subclasei *PP1* este complet determinat. Se observă uşor că toţi parametrii sistemului de aşteptare pot fi exprimaţi ca funcţii de λ şi μ.

Referitor la λ şi μ facem următoarele observaţii:

a) dacă $\lambda > \mu$ firul de aşteptare creşte nelimitat deoarece fluxul de sosire este superior fluxului de servire. În acest caz sistemul nu poate intra în regim staţionar;

b) dacă $\lambda = \mu \Rightarrow \rho = 1$ şi din formula (9—27) rezultă că $\bar{n}_f \to \infty$;

c) dacă $\lambda < \mu$ firul de aşteptare va avea o lungime finită şi după un oarecare timp sistemul va intra în regim staţionar.

Scanned with OKEN Scanner

În mod asemănător, însă pe baza unui calcul laborios, se stabileşte modelul matematic al clasei PPS, model descris de următoarele formule:

$$P_0(t) = \left(\frac{\rho^S}{S!(1-\rho/S)} + \sum_{n=0}^{S-1}\frac{\rho^n}{n!}\right)^{-1} \tag{9—28}$$

$$P_n(t) = \begin{cases} \frac{\rho^n}{n!} P_0(t) \text{ dacă } 1 \leqslant n < S \\ \frac{\rho^n}{S^{n-S}S!} P_0(t) \text{ dacă } n \geqslant S \end{cases} \tag{9—29}$$

$$n_s = \left[\rho + 2\frac{\rho^2}{2!} + \ldots + S\frac{\rho^S}{S!} + \frac{(S+1)\rho^{S+1}}{S!(S-\rho)} + \frac{S+2}{S!(S-\rho)^2}\right] \cdot P_0(t), \tag{9—30}$$

$$\bar{n}_f = \frac{\rho^{S+1}}{S.S!(1-\rho/S)^2} P_0(t), \tag{9—31}$$

$$P(t) = \frac{\rho^S}{S!(1-\rho/S)} P_0(t), \tag{9—32}$$

$$\bar{t}_f = \frac{\rho^S}{S.S!(1-\rho/S)^2\mu} P_0(t), \tag{9—33}$$

$$\bar{t}_s = \bar{t}_f + \frac{1}{\mu}. \tag{9—34}$$

În cazul în care $S \to \infty$, toţi clienţii care sosesc sînt imediat serviţi. Deci în cazul subclasei PPS cu $S \to \infty$ nu se pune problema formării unui şir de aşteptare.

Cu aceasta, modelele matematice a clasei PPS^* cu disciplina primul sosit-primul servit a sistemelor fără pierderi şi staţii de servire paralele, sînt complet descrise. Rămîne să abordăm, acum, aspectul economic şi anume modul cum se stabileşte „compromisul" între costul aşteptării şi costul serivrii.

Fie c_1 costul unităţii de timp a unui client şi c_2 costul unei unităţi de timp a unei staţii de servire. Dacă T este intervalul de timp în care dorim să calculăm costul mediu al sistemului c_s, atunci timpul pierdut de clienţi în intervalul T este $\lambda\bar{t}_f T$, iar costul va fi $c_1\lambda\bar{t}_f T$ şi $(S-\rho)T$ va fi timpul total mediu de nefolosire a staţiilor cu costul $c_2(S-\rho)T$. De aici rezultă costul mediu al sistemului:

$$c_s = [c_1\lambda\bar{t}_f + c_2(S-\rho)]T. \tag{9—35}$$

Se caută să se minimizeze acest cost c_s, deci să se minimizeze timpul de inactivitate a staţiilor.

Pentru determinarea soluţiei optime se calculează c_s cu diferite valori pentru S în modul următor:

$$c_{s0} = \min_{i \in N}\{[c_1\lambda\bar{t}_f + c_2(i-\rho)]T\}. \tag{9—36}$$

Scanned with OKEN Scanner

Se determină, în felul acesta, numărul de staţii $S = i$ în aşa fel încît costul să fie minim.

Pentru a studia un sistem de aşteptare indicăm următoarele etape:

Etapa I. Determinăm perioadele de timp în care sistemul poate fi considerat staţionar.

Etapa a II-a. Determinăm, prin măsurători de timp (cronometrare), parametrii λ şi μ în perioadele în care sistemul este presupus staţionar.

Etapa a III-a. Verificăm prin intermediul testului χ^2 presupunerea făcută asupra naturii poissoniene a sosirilor şi servirilor.

Etapa a IV-a. Calculăm parametrii ρ, $P_n(t)$, $P_0(t)$, $P(t)$, $\bar{n}_s$, $\bar{n}_f$, $\bar{t}_f$, $\bar{t}_s$ şi $\bar{\rho}$ prin intermediul formulelor (9—20,*a*), ..., (9—27) sau (9—28), ..., (9—34).

Etapa a V-a. Se caută să se optimizeze costul global al sistemului, ţinînd cont de preţul unitar al aşteptării clienţilor şi al aşteptării staţiilor, folosind formulele (9—35) şi (9—36).

Ca aplicaţie, vom studia un sistem de aşteptare făcînd calculul manual, apoi vom studia acelaşi sistem cu ajutorul programului 'PPSLI' şi a unui calculator automat ('PPSLI' este scris în limbajul FORTRAN şi este utilizabil în sistemele de calcul FELIX C—256).

Un exemplu de aplicaţie

Într-o întreprindere pentru construcţii şi/sau reparaţii de maşini şi utilaje petroliere, aprovizionarea atelierelor cu materiale şi piese de schimb se face (să presupunem) de la o magazie ce are un singur ghişeu de distribuire.

Şeful de depozit, observînd marea afluenţă la ghişeu, cere înfiinţarea unui nou ghişeu, deci înfiinţarea unui nou post de distribuitor-materiale.

Conducerea întreprinderii, înainte de a lua o hotărîre, cere unui analist de la oficiul de calcul să facă un studiu de oportunitate, spre a se vedea dacă este necesară înfiinţarea unui nou post. Analistul, observînd afluenţa muncitorilor la ghişeu mai multe zile în şir, ajunge la concluzia că în primele 30—45 minute ale unui schimb, precum şi în timpul pauzei de masă (circa 1 oră), prezenţa muncitorilor la ghişeu este slabă şi chiar foarte slabă. În restul timpului însă se formează cozi (fire de aşteptare).

Problema a fost rezolvată de analist în etape, precum urmează:

Etapa I. Stabilirea perioadelor presupuse staţionare.

Analistul presupune că fenomenul de aşteptare, descris mai sus, este staţionar în perioadele în care prezenţa muncitorilor la ghişeu este mare.

Etapa a II-a. Determinarea pe bază de cronometrări a parametrilor λ şi μ.

Este indicat ca atunci cînd se aplică testul de semnificaţie χ^2, numărul de elemente ale mulţimii (populaţiei statistice) ce se cercetează să fie în general mai mare decît 50. În cazul de faţă, s-au luat în considerare cîte 80 intervale de 5 minute pentru studiul sosirilor, intervale situate în perioada presupusă staţionară (intervalele pot fi consecutive sau nu) şi s-au numărat muncitorii sosiţi în cursul fiecărui interval. S-au obţinut datele din tabela 9—2.

Scanned with OKEN Scanner

Tabela 9—2

Nr. crt. (n)	Număr muncitori (m_n) sosiți în timpul unui interval de 5 minute	Frecvență observată (f_n)	$m_n \cdot f_n$
1	0	25	0
2	1	30	30
3	2	17	34
4	3	5	15
5	4	2	8
6	5	1	5
7	6	0	0
—	—	$N = \sum_{n=1}^{7} f_n = 80$	$\sum_{n=1}^{7} m_n f_n = 92$

Calculînd media distribuției sosirilor avem :

$$M(X) = \sum_{n=1}^{7} \frac{m_n f_n}{N} = \frac{92}{80} = 1{,}15.$$

Dar $M(X) = \lambda t$ și cum $t = 5$ minute rezultă :

$$\lambda = \frac{1{,}15}{5} = 0{,}23 \text{ sosiri/min.}$$

Pentru studiul servirilor s-au observat duratele servirilor în 80 de cazuri din perioada presupusă staționară. S-au obținut datele din tabela 9—3.

Tabela 9—3

Nr. crt. (n_1)	Durata unui serviciu ($t_1 - t_2$)	Număr de serviri observate (f_{n1})	$\Delta t_{n1} = \frac{t_2 - t_1}{2}$	$\Delta t_{n1} \cdot f_{n1}$
1	sub 3 minute	33	1,5	49,5
2	de la 3 la 4 minute	18	3,5	63,0
3	4— 5	7	4,5	31,5
4	5— 6	6	5,5	33,0
5	6— 7	4	6,5	26,0
6	7— 8	5	7,5	37,5
7	8— 9	2	8,5	17,0
8	9—10	1	9,5	9,5
9	10—11	2	10,5	21,0
10	11—12	1	11,5	11,5
11	12—13	1	12,5	12,5
12	13—14	0	13,5	0
—	—	$N_1 = \sum_{n_1=1}^{12} f_{n1} = 80$	—	$\sum_{n_1=1}^{12} t_{n1} \times f_{n1} = 312$

Deoarece servirile sînt înregistrate în intervale de timp $[t_1, t_2]$, vom considera fiecare interval concentrat în valoarea medie a sa $\Delta t = \frac{t_2 - t_1}{2}$,

Scanned with OKEN Scanner

valori aflate în coloana a patra a tabelei 9—3. Din această tabelă deducem:

$$\frac{1}{\mu} = \frac{\sum_{n1=1}^{12} t_{n1} \times f_{n1}}{N_1} = \frac{312}{8} = 3{,}9 \text{ şi } \mu = \frac{1}{3{,}9} = 0{,}25641.$$

Etapa a III-a constă în verificarea ipotezei că sosirile şi servirile sînt poissoniene.

Pentru a verifica această ipoteză vom folosi testul χ^2. Pentru aceasta se întocmeşte tabela 9—4.

Tabela 9—4

Număr sosiri (n)	Frecvenţa observată (f_n)	Frecvenţa teoretică (f_{tn})	$\Delta n = t_{tn} - f_n$	Δ_n^2	Δ_n^2/f_{t_n}
0	25	25,33120	0,33120	0,109693	0,004330
1	30	29,13088	0,86912	0,755370	0,025930
2	17	16,75026	0,24974	0,062370	0,003724
3	5	6,42093	1,42093	2,019042	0,314447
$\geqslant 4$	3	2,36673	0,63327	0,401030	0,169445
—	$N = \sum_{n=0}^{4} f_n = 80$	$\sum_{n=0}^{4} f_{t_n} = 80$	—	$\chi_c^2 = \sum_{n=0}^{4} \frac{\Delta n^2}{f_{t_n}} = 0{,}517876$	

Calculul frecvenţei teoretice f_{t_n} se face folosind relaţia:

$$f \quad N \frac{(\lambda t)^n}{n!} e^{-\lambda t},$$

relaţie ce poate fi transformată într-o relaţie de recurenţă, adică:

$$f_{t_n} = N \frac{(\lambda t)^n}{n!} e^{-\lambda t} = \frac{\lambda t}{n} \frac{(\lambda t)^{n-1}}{(n-1)!} N e^{-\lambda t} = \frac{\lambda t}{n} f_{t_{n-1}}.$$

Vom folosi această relaţie pentru $\forall n \in \{0, n-1\}$, iar pentru $n := n$ vom calcula f_{t_n} cu formula:

$$f_{t_n} = N - \sum_{n=0}^{n-1} f_{t_n}.$$

Observăm că ultimele patru linii (clase) ale tabelei 9—2 au fost restrînse în tabela 9—4 într-o singură linie (clasă) pentru $n \geqslant 4$. Deoarece $e^{-\lambda t} = e^{-1{,}15} \approx 0{,}31664$, cu ajutorul formulei de recurenţă se poate întocmi uşor, pe bază de calcul, tabela 9—4.

Din ultima coloană a tabelei 9—4 se obţine valoarea lui χ_c^2. Folosind tabelele uzuale cu valorile lui χ^2, în funcţie de probabilitatea P cu care dorim ca fenomenul să fie considerat poissonian şi de grade de libertate ν, se extrage χ_0^2:

1° dacă $\chi_0^2 > \chi_c^2$, atunci ipoteza asupra naturii poissoniene a sosirilor se verifică cu probabilitatea P;

Scanned with OKEN Scanner

2° dacă $\chi_0^2 < \chi_c^2$, atunci ipoteza asupra naturii poissoniene a sosirilor nu este verificată.

În cazul nostru, analistul a hotărît că dacă testul χ^2 se verifică cu $P = 0{,}90$ atunci va admite că sosirile sînt poissoniene. În fond, admite că 10% din sosiri pot să se producă și după o altă lege. Pentru a determina numărul de grade de libertate ν se folosește formula:

$$\nu = c - 1 - p,$$

unde: c este numărul de clase (în cazul acestei aplicații $c = 5$), iar p este numărul de parametri ($p = 1$); atunci:

$$\nu = 5 - 1 - 1 = 3.$$

Se găsește în tabelele uzuale, pentru $P = 0{,}90$ și $\nu = 3$: $\chi_0^2 = 0{,}584$. Deoarece în cazul acestui exemplu (v. tabela 9—4), $\chi_c^2 = 0{,}5179$ înseamnă că $\chi_0^2 > \chi_c^2$ și ipoteza asupra naturii poissoniene a sosirilor este verificată.

Pentru a verifica că duratele de servire urmează o repartiție exponențială (deci servirile sînt poissoniene) vom folosi testul lui Kolmogorov*.

Pentru verificarea concordanței dintre repartiție empirică (observată) și cea teoretică prin intermediul testului lui Kolmogorov vom proceda în felul următor:

1° calculăm frecvențele relative cumulate observate f_{rc} și frecvențele cumulate teoretice f_{rct} cu ajutorul formulelor:

$$f_{rc} = \frac{f_{cn_1}}{N_1} \text{ și } f_{rct} = 1 - e^{-\mu \Delta t_{n1}};$$

2° calculăm $\Delta n_1 = |f_{rc} - f_{rct}|$;

3° calculăm $D = \max\limits_{n_1 = 1, n_1} (\Delta n_1)$ și cu ajutorul relației $\lambda_c = D\sqrt{N_1}$ se calculează λ_c;

4° dintr-o tabelă cu probabilitățile:

$$P(\lambda_t) = \lim_{n \to \infty} \left\{ \sup_{-\infty < \Delta t < +\infty} |f_{rc}(\Delta t) - f_{rct}(\Delta t)| \leqslant \frac{\lambda_t}{\sqrt{n}} \right\},$$

extragem pe λ_t corespunzător probabilității dorite.

Dacă $\lambda_c < \lambda_t$ se poate afirma că repartiția duratelor de servire observate urmează o lege exponențială; dacă nu, ipoteza se respinge și eventual se caută o altă repartiție teoretică.

În cazul acestui exemplu, pentru aplicarea testului lui Kolmogorov, vom completa o nouă tabelă (tabela 9—5), care conține și valorile e^{-x}.

Din ultima coloană a tabelei 9—5 găsim $\Delta = 0{,}093232$ și deci $\lambda_c = 0{,}093232 \cdot \sqrt{80} = 0{,}833892$.

Dintr-o tabelă cu valobile lui $P(\lambda_t)$, pentru $P(\lambda_t) = 0{,}90$ găsim $\lambda_t = 1{,}223558$. Comparînd λ_t și λ_c se observă că $\lambda_c < \lambda_t$ și deci repartiția duratelor de servire urmează o lege exponențială.

Etapa a IV-a se referă la calculul parametrilor sistemului de așteptare.

---

* S-ar putea folosi, totuși, în același mod și testul χ^2.

Scanned with OKEN Scanner

Pentru a calcula parametrii sistemului de așteptare vom utiliza relațiile (9—20,*a*), ..., (9—26). Obținem:

$$\rho = \frac{\lambda}{\mu} = 0{,}897,$$

$$P_0 = 1 - \rho = 0{,}103,$$

$$\bar{n}_{S1} = \frac{\rho}{1-\rho} = 8{,}71 \text{ și deci în sistem avem } \approx 9 \text{ muncitori},$$

$$\bar{n}_{f1} = \frac{\rho^2}{1-\rho} = \frac{0{,}8046}{0{,}103} = 7{,}81 \Rightarrow \text{există circa 8 muncitori în așteptare},$$

$$\bar{t}_{s1} = \frac{1}{\mu - \lambda} = 37{,}86 \text{ min},$$

$$\bar{t}_{f1} = \frac{\rho}{\mu - \lambda} = \frac{0{,}897}{0{,}02641} = 33{,}96 \text{ min},$$

$$\bar{\rho} = S - \rho = 0{,}103.$$

Constatăm că timpul pierdut de muncitori cu așteptarea este suficient de mare; cu toate acestea, se poate constata că distribuitorul are o perioadă de inactivitate $t_{inact} = 8.60 \cdot \rho = 50$ min în fiecare schimb. Cum timpul pierdut în medie de un muncitor este considerabil, se va trece la optimizarea sistemului de așteptare în vederea eliminării timpului pierdut de muncitori, ceea ce va duce la un cîștig la timpul de muncă, implicit la o creștere a producției.

Etapa a V-a constă în optimizarea sistemului de așteptare.

Am arătat mai sus că una din căile de optimizare a unui sistem de așteptare se realizează prin încercări, dînd lui S valorile 1, 2, 3, ... și calculînd costul timpului pierdut în sistem și eventual și toți parametrii (sau numai o parte din ei).

Pentru $S = 1$ sînt calculați deja toți parametrii. Urmează să calculăm costul timpului pierdut în sistem. Considerînd retribuție orară pentru muncitor: $c_1 = 7{,}75$ lei/h și pentru distribuitor: $c_2 = 6{,}25$ lei/h, atunci costul mediu al timpului pierdut în sistem va fi:

$$c_{s_1} = [c_1 \lambda \bar{t}_f + c_2(S - \rho)]T = \left(\frac{7{,}75}{60} \times 0{,}23 \times 33{,}96 + \frac{6{,}25}{60} \times 0{,}103\right) \cdot$$

$$\cdot 480 \simeq 489{,}40 \text{ lei}.$$

Pentru $S = 2$

$$P_0 = \left(\frac{\rho S}{S!\left(1 - \frac{\rho}{S}\right)} + \sum_{n=0}^{S-1} \frac{\rho^n}{n!}\right)^{-1} = \frac{1}{\left[\frac{0{,}897^2}{2!\left(1 - \frac{0{,}897}{2}\right)} + 1 + 0{,}897\right]} =$$

$$= 0{,}380739;$$

Scanned with OKEN Scanner

Tabela 9—5

Nr. crt. (n)	Δt_{n_1}	Frecvența observată (f_{n_1})	Frecvența cumulată observată (f_{cn_1})	Frecvența relativă cumulată $\left(f_{nc} = \frac{f_{cn_1}}{N_1}\right)$	$\mu\Delta t_{n_1}$	$e^{-\mu\Delta t_{n_1}}$	Frecvența relativă teoretică cumulată ($f_{rct} = 1 - e^{-\mu\Delta t_{n_1}}$)	$\Delta_{n_1} = \lvert f_{rc} - f_{rct} \rvert$
1	1,5	33	33	0,4125	0,3846	0,680732	0,319268	0,093232
2	3,5	18	51	0,6375	0,8974	0,407633	0,592367	0,045133
3	4,5	7	58	0,7250	1,1538	0,315443	0,684557	0,040443
4	5,5	6	64	0,8000	1,4103	0,244067	0,755933	0,044067
5	6,5	4	68	0,8500	1,6667	0,189080	0,810920	0,039080
6	7,5	5	73	0,9125	1,9231	0,146283	0,853717	0,058783
7	8,5	2	75	0,9375	2,1795	0,113190	0,886810	0,050690
8	9,5	1	76	0,9500	2,4359	0,087618	0,912382	0,037618
9	10,5	2	78	0,9750	2,6923	0,67754	0,932246	0,042754
10	11,5	1	79	0,9875	2,9487	0,052473	0,947527	0,039973
11	12,5	1	80	1,0000	3,2051	0,040562	0,959438	0,040562
—	—	$N_1 = \sum f_{n_1} = 80$	—	—	—	—	—	$D = max(\Delta_{n_1}) = 0,093232$

Scanned with OKEN Scanner

$$\bar{n}_{f2} = \frac{P_0 \cdot \rho^3}{S \cdot S!(1 - \rho/S)^2} = \frac{0{,}380739 \cdot 0{,}897^3}{2 \cdot 2\left(1 - \frac{0{,}897}{2}\right)^2} = 0{,}226;$$

$$\bar{n}_{S_2} = \left[\rho + 2\frac{\rho^2}{2!} + \ldots + S\frac{\rho^S}{S!} + \frac{(S+1)\rho^{S+1}}{S!(S-\rho)} + \frac{S+2}{S!(S-\rho)}\right] \cdot P_0 =$$

$$= \left[0{,}897 + 0{,}897^2 + \frac{3 \cdot 0{,}897^3}{2!(2 - 0{,}897)} + \frac{4}{2!(2 - 0{,}897)^2}\right] \cdot 0{,}380739 = 1{,}648;$$

$$\bar{t}_{f_2} = \frac{P_0 \rho^2}{S \cdot S!\left(1 - \frac{\rho}{S}\right)^2 \cdot \mu} = \frac{0{,}380739 \cdot 0{,}897^2}{2 \cdot 2!\left(1 - \frac{0{,}897}{2}\right)^2 \cdot 0{,}25641} = 0{,}982;$$

$$\bar{t}_{S_2} = \bar{t}_{f_2} + \frac{1}{\mu S} \simeq 2{,}93 \text{ min};$$

$$\bar{\rho} = S - \rho = 1{,}103.$$

Costul mediu al timpului pierdut în sistemul de aşteptare va fi deci:

$$c_{S_2} = [c_1 \lambda \bar{t}_{f_1} + c_2(S - \rho)]T = (7{,}75 \times 0{,}23 \times 0{,}982 + 6{,}25 \times 1{,}103) \cdot 8 \simeq 69{,}15 \text{ lei}.$$

Pentru $S = 3$ nu vom mai calcula toţi parametrii sistemului, deoarece în cazul problemei pe care o avem de rezolvat (minimizarea costului mediu al timpului pierdut) sînt suficienţi P_1 şi $\bar{t}_{f_3}$. Deci:

$$P_0 = \left[\frac{0{,}897^3}{3!\left(1 - \frac{0{,}897}{3}\right)} + 1 + 0{,}897 + \frac{0{,}897^2}{2!}\right]^{-1} = \frac{1}{2{,}470901} = 0{,}404711,$$

$$\bar{t}_{f_3} = \frac{0{,}897^3 \cdot 0{,}404711}{3 \cdot 3!\left(1 - \frac{0{,}897}{3}\right)^2 \cdot 0{,}25641} = 0{,}129.$$

Costul mediu al timpului pierdut în sistem va fi

$$c_{S_3} = [c_1 \lambda \bar{t}_{f_3} + c_2(S - \rho)]T = (7{,}75 \times 0{,}23 \times 0{,}129 + 6{,}25 \times 2{,}103) \times 8 \simeq 107 \text{ lei}.$$

Observăm că dacă se vor angaja încă 2 distribuitori pe schimb, costul mediu al timpului pierdut va fi 107 lei, cost mai mare decît dacă se angajează un singur distribuitor.

Deci, soluţia optimă este să se angajeze încă un distribuitor de materiale la fiecare schimb. Prin această optimizare întreprinderea va avea de cîştigat $489{,}40 - 69{,}15 = 420{,}25$ lei pe schimb, la care se poate adăuga şi sporul de producţie care se realizează de către muncitorii care nu se mai află în firul de aşteptare. Pentru fiecare muncitor se cîştigă un timp productiv $t_p = \bar{t}_{s_1} - \bar{t}_{s_2} \simeq 35$ minute. Se poate trage deci concluzia că angajarea unui distribuitor de materiale în fiecare schimb este rentabilă pentru întreprindere.

Scanned with OKEN Scanner

Pentru rezolvarea aceleiași probleme într-un timp foarte scurt și cu exactitate sporită am realizat un program original denumit de noi PPSLI, program scris în limbaj FORTRAN și utilizabil la calculatoare FELIX C—256.

Programul este prezentat în lista 9.1, iar schema logică (organigrama) în figura 9.4.

Semnificația variabilelor și constantelor cu rol important, folosite în programul PPSLI, este următoarea:

FN — vector ce conține frecvența observată, atît pentru sosiri cît și pentru serviri;

FTN — vectorul frecvență teoretică pentru sosiri;

HI — matrice conținînd valorile testului χ^2. Coloanele semnifică valori-probabilitate cu următoarea corespondență:

nr. coloană 1 2 3 4 5 6 7 8 9 10 11 12 13

probabilitate

0,99 0,98 0,95 0,90 0,80 0,70 0,50 0,30 0,20 0,10 0,05 0,02 0,01

iar liniile sînt corespunzătoare lui $\nu = 1,30$;

TKO — matrice conținînd valorile testului Kolmogorov. Prima linie conține 143 valori pentru $P(\lambda)$, iar cea de a doua linie valorile corespunzătoare pentru λ. Valorile lui *HI* și *TKO* se tipăresc după citire, pentru verificarea mai ușoară a corectitudinii valorilor introduse;

FRC — vectorul frecvență relativă cumulată observată;

FRT — vectorul frecvență cumulată teoretică;

MN — vectorul ce conține numărul de sosiri într-un interval de timp, sau limita superioară a intervalului de timp al servirilor;

CS — vectorul ce conține costul mediu al timpului pierdut în sistem pentru $S = 1, 2, \ldots$;

N2 (în cazul citirii de pe cartelă) — numărul de cazuri luate în studiu, atît la sosiri cît și la serviri;

NCL — numărul de clase în care se vor regrupa cazurile luate în studiu;

NP — numărul de parametrii în cazul sosirilor;

P — număr întreg ce reprezintă corespondentul probabilității din tabela χ^2. Spre exemplu, pentru probabilitatea 0,50, $P = 7$ în cazul testului χ^2 sau pentru probabilitatea 0,1357, $P = 27$ în cazul servirilor;

NT — mărimea intervalului de timp exprimat în minute, în care se fac observațiile în cazul sosirilor;

NS — numărul stațiilor de servire;

C1 — retribuția orară a unui muncitor;

C2 — retribuția orară a unui distribuitor-materiale;

N — un număr natural $\neq 0$, utilizat ca indice al lui P_n.

Programul calculează parametrii sistemului de așteptare și costul mediu al timpului pierdut în sistem. Se simulează apoi mărirea numărului de stații de servire a sistemului, $S := S + 1$ pînă cînd $c_{s_{i+1}} < c_{si}$, unde c_{si} este costul timpului mediu al timpului pierdut inițial. Cartelele de date

Scanned with OKEN Scanner

```
C        LISTA 9.1
C      ***********************************************************************
C      * PROGRAM PPSLI.                                                      *
C      * PROGRAMUL  SE FOLOSESTE  PENTRU  STUDIEREA  SISTEMELOR  DE  AS-*
C      * TEPTARE  IN CARE  SOSIRILE  SINT  PRESUPUSE  POISSONIENE  SI    *
C      * INTERVALELE  DE TIMP  IN  CARE  SE  EXECUTA  SERVIRILE  URMIND *
C      * O LEGE  EXPONENTIALA, IN  PROGRAM  SE  VERIFICA  ACESTE  PRESU-*
C      * PUNERI  SI  DACA  SINT  SATISFACUTE  SE  CALCULEAZA  PARAMETRII*
C      * SISTEMULUI  SI  PRIN  SIMULARI  SE  OPTIMIZEAZA(MINIMIZEAZA)    *
C      * COSTUL  MEDIU  AL  TIMPULUI  PIERDUT  IN  SISTEM                *
C      ***********************************************************************
       DIMENSION FN(50),FTN(50),HI(30,13),TKO(2,143),FRC(50),FRT(50)
       REAL LAMDA,MIU,LAMDAT,MN(50),LAMDAC,CS(10),NSB,NFB
       INTEGER P
       CALL TESTX(HI)
       CALL TESTK(TKO)
     2 READ(105,105,END=13) N2,NCL,NP,P,NT
   105 FORMAT(5I2)
       READ(105,106)(MN(I),FN(I),I=1,N2)
   106 FORMAT(20(2F2.0))
       N3=1
       A=SUM(N3,FN,MN,N2)
       N3=0
       B=SUM(N3,FN,MN,N2)
       LAMDAT=B/A
       N3=NCL+1
       DO 3 I=N3,N2
     3 FN(NCL)=FN(NCL)+FN(I)
       NIU=NCL-NP-1
       NN=NCL-1
       FTN(1)=EXP(-LAMDAT)*A
       DO 4 I=2,NN
     4 FTN(I)=FTN(I-1)*LAMDAT/(I-1)
       N3=1
       FTN(NCL)=A-SUM(N3,FTN,MN,NN)
       HIC=0
       DO 5 I=1,NCL
       DELTA=FN(I)-FTN(I)
       DELTA=DELTA**2
     5 HIC=HIC+DELTA/FTN(I)
       WRITE(108,107) NT,(I,MN(I),I,FN(I),I,FTN(I),I=1,NCL)
   107 FORMAT(1H1,///,1H ,25X,'VERIFICAREA  IPOTEZEI  FLUX  DE  SOSIRE  P
      1OISSON'/1H ,25X,48(1H*)//1H ,5X,'NUMAR DE SOSIRI IN ',I2,' MINUTE'
      2,5X,'FRECVENTA OBSERVATA ',16X,'FRECVENTA TEORETICA'/(1H ,5X,'NM('
      3,I2,')=',F10.4,16X,'FN(',I2,')=',F10.4,19X,'FNT(',I2,')=',F10.4))
       IF(HIC.LT.HI(NIU,P)) GO TO 6
       WRITE(108,108) A,LAMDAT,HIC,HI(NIU,P)
   108 FORMAT(//1H ,20X,'N=',F10.4,5X,'LAMDAT=',F10.4,5X,'HIC=',F7.4,5X,'
      1HI=',F7.4//1H ,20X,'IPOTEZA  NEVERIFICATA'/1H ,20X,21(1H*)//)
       GO TO 13
     6 LAMDA=LAMDAT/NT
       WRITE(108,109) A,LAMDAT,LAMDA,HIC,HI(NIU,P)
   109 FORMAT(//1H ,10X,'N=',F10.4,3X,'LAMDAT=',F10.4,3X,'LAMDA=',F10.4,3
      1X,'HIC=',F7.4,3X,'HI=',F7.4//1H ,10X,'IPOTEZA  ASUPRA  SOSIRILOR
      2ESTE  VERIFICATA'/1H ,10X,44(1H-))
       READ(105,110)N2,NCL,P,NS,C1,C2,N
   110 FORMAT(2I2,I3,I2,2F6.2,I2)
       READ(105,111)(MN(I),FN(I),I=1,N2)
   111 FORMAT(10(2F3.0))
       A=0
       DO 7 I=1,N2
       R=(MN(I)+A)/2.
       A=MN(I)
```

STOP

```
      MN(I)=R
      N3=1
      A=SUM(N3,FN,MN,N2)
      N3=0
      B=SUM(N3,FN,MN,N2)
      MIU=A/B
      A1=0
      D=0
      DO 8 I=1,NCL
      FN(I)=FN(I)+A1
      FRC(I)=FN(I)/A
      T=MIU*MN(I)
      FRT(I)=1-EXP(-T)
      DIF=ABS(FRT(I)-FRC(I))
      D=AMAX1(D,DIF)
    8 A1=FN(I)
      WRITE(108,112)(I,MN(I),I,FN(I),I,FRC(I),I,FRT(I),I=1,NCL)
   12 FORMAT(1H1,///1H ,25X,'VERIFICAREA IPOTEZEI CA SERVIRILE URMEA
     1ZA O LEGE EXPONENTIALA'/1H ,25X,68(1H*)//1H ,17X,'VALORI MEDII
     2A',8X,'FRECVENTA CUMULATA  FRECVENTA RELATIVA  FRECVENTA RELAT
     3IVA'/1H ,15X,'INTERVALELOR DE TIMP',8X,'OBESEVATA',14X,'CUMULATA
     49X,'TEORETICA CUMULATA'//(1H ,17X,'DTN1(',I2,')=',F7.3,9X,'FCN(',I
     52,')=',F4.0,7X,'FRC(',I2,')=',F8.6,6X,'FRCT(',I2,')=',F8.6))
      LAMDAC=D*SQRT(A)
      LAMDAT=TKO(2,P)
      IF(LAMDAC.LT.LAMDAT) GO TO 9
      WRITE(108,113) D,LAMDAC,LAMDAT,MIU
  113 FORMAT(///1H ,15X,'D=',F9.6,10X,'LAMDAC=',F9.6,5X,'LAMDAT=',F9.6,
     15X,'MIU=',F8.6/1H ,25X,'IPOTEZA  NU SE VERIFICA'/1H ,25X,23(1H*))
      GO TO 13
    9 WRITE(108,114)D,LAMDAC,LAMDAT,MIU
  114 FORMAT(///1H ,15X,'D=',F9.6,10X,'LAMDAC=',F9.6,5X,'LAMDAT=',F9.6,
     15X,'MIU=',F8.6/1H ,25X,'IPOTEZA SE VERIFICA'/1H ,25X,19(1H-))
      DO 12 I=1,10
   12 CS(I)=0
   13 RO=LAMDA/MIU
      WRITE(108,115) LAMDA,MIU,RO
  115 FORMAT(1H1,////1H ,30X,'STUDIUL SISTEMULUI DE ASTEPTARE'/1H ,30X,3
     11(1H=)////1H ,15X,'LAMDA=',F8.5,5X,'MIU=',F8.5,5X,'RO=',F8.5//)
      IF(NS.NE.1) GO TO 10
      PN=RO**N*(1-RO)
      P0=1-RO
      ROB=NS-RO
      PNUL=RO
      NSB=RO/P0
      NFB=RO*RO/P0
      TF=RO/(MIU-LAMDA)
      TS=1./(MIU-LAMDA)
      NIN=1
      CS(1)=(C1*LAMDA*TF+C2*ROB)*8
      WRITE(108,117) NIN,P0,PNUL,N,PN,NSB,NFB,TF,TS,ROB,CS(NIN)
      NIN=NIN+1
      GO TO 11
   10 NIN=NS
   11 A=1
      DO 14 I=1,NIN
   14 A=(A*RO)/I
      R=A/(1-RO/NIN)
      L=NIN-1
      B=0
      B1=1
      DO 15 I=1,L
      B1=(B1*RO)/I
```

Scanned with OKEN Scanner

```
      15 B=B+B1
         P0=1./(R+B+1)
         PNUL=R*P0
         NFB=PNUL*RO/(NIN*(1-RO/NIN))
         TF=PNUL/(NIN*(1-RO/NIN)*MIU)
         TS=TF+1./(MIU*NIN)
         ROB=NIN-RO
         NSB=KO
         A2=RO
         L=NIN-1
         DO 16 I=1,L
         A2=A2*RO/I
      16 NSB=NSB+A2
         A1=A*(NIN+1)*RO/(NIN-KO)
         A2=(NIN+2)/(FACT(NIN)*(NIN-KO)**2)
         NSB=(NSB+A1+A2)*P0
         IF(N.GE.NIN) GO TO 18
         PN=1
         DO 19 I=1,N
      19 PN=PN*RO/I
         PN=PN*P0
         GO TO 20
      18 PN=RO**NIN/(NIN**(N-NIN)*FACT(NIN))
      20 CS(NIN)=(C1*LAMDA*TF+C2*ROB)*8
         IF(CS(NIN).GT.CS(NS)) GO TO 2
         WRITE(108,117) NIN,P0,PNUL,N,PN,NSB,NFB,TF,TS,ROB,CS(NIN)
     117 FORMAT(1H ,15X,2HS=,I2,3X,3HP0=,F6.4,3X,2HP=,F6.4,3X,2HN=,I2,3X,3H
    1PN=,F6.4,3X,3HNS=,F7.4,3X,3HNF=,F7.4,3X,3HTF=,F7.4,3X,3HTS=,F7.4,3
    2X,4HROB=,F7.4/1H ,15X,5(1H=)/1H ,15X,'COSTUL MEDIU AL TIMPULUI PIE
    3RDUT IN SISTEM ESTE:  CS=',F8.2/1H ,65X,11(1H=)/)
         NIN=NIN+1
         GO TO 11
      13 STOP
         END
```

54010001 29/10/75 15.46.44

```
      FUNCTION SUM(N,FN,MN,N2)
C     ************************************************************************
C     *                                                                      *
C     * FUNCTIA  SUM.                                                        *
C     * FUNCTIA  CALCULEAZA SUMA  FN(1)+ FN(2)+...+FN(N2)  DACA  N=1         *
C     * FUNCTIA  CALCULEAZA SUMA  FN(1)* MN(1)+...+FN(N2)*MN(N2)  DACA       *
C     * N ESTE  DIFERIT  DE 1.                                               *
C     ************************************************************************
      REAL FN(50),MN(50),L
      S=0
      DO 1 I=1,N2
      IF(N.EQ.1) GO TO 2
      L=MN(I)
      GO TO 1
    2 L=1
    1 S=S+FN(I)*L
      SUM=S
      RETURN
      END
```

```
      SUBROUTINE TESTK(TKO)
C     ************************************************************************
C     * SE INTRODUC IN MEMORIE VALORILE NECESARE APLICARII TESTULUI          *
C     * KOLMOGOROV. SE FOLOSESTE TABELA DE LA PAG. 389 DIN TEORIA ERO-       *
C     * RILOR DE MASURARE SI METODA CELOR MAI MICI PATRATE, M.TIRON,         *
C     * EDITURA TEHNICA BUCURESTI 1972. TIPARIREA DATELOR CITITE SE          *
C     * FACE CONDITIONAT. SE INTRODUC IN PRIMUL RIND VALORILE PROBABI-       *
C     * LITATII, CITE 13 VALORI CONSECUTIVE PE O CARTELA, APOI VALORILE      *
C     * CORESPONDENTE PENTRU LAMDA.SE FOLOSESC NUMAI 143 VALORI ADICA        *
C     * PINA LA PROBABILITATEA P(LAMDA)=0.9959                               *
C     ************************************************************************
```

```
      DIMENSION TKO(2,143)
      READ(105,102) ((TKO(I,J),J=1,143),I=1,2)
  102 FORMAT(13F5.4)
C     WRITE(108,103)
C103 FORMAT(1H1,///1H ,45X,'VALORI  TEST  KOLMOGOROV',/1H ,45X,24(1H*)/
C    1 1H ,4X,126(1H*))
C     N1=1
C     N2=13
C  1 WRITE(108,104) ((TKO(I,J),J=N1,N2),I=1,2)
C104 FORMAT(1H ,4X,'* P(LT) *',13(1X,F6.4,2H *)/1H ,4X,126(1H*)/1H ,4X,
C    1'*   LT  *',13(1X,F6.4,2H *)/1H ,4X,126(1H*))
C     N1=N1+13
C     N2=N2+13
C     IF(N2.LE.143)GO TO 1
      RETURN
      END
```

```
      FUNCTION FACT(N)
C     ***************************************************************
C     * FUNCTIA  FACT.                                              *
C     * FUNCTIA  CALCULEAZA  FACTORIALUL  DE N. FACT=N!             *
C     ***************************************************************
      K=1
      DO 1 I=1,N
    1 K=K*I
      FACT=K
      RETURN
      END
```

```
      SUBROUTINE TESTX(HI)
C     ***************************************************************
C     * SE INTRODUC IN MEMORIE VALORILE TESTULUI XI**2 SI SE TIPARESC *
C     * CONDITIONAT, DE REGULA TIPARIREA SE VA FACE LA O PRIMA RULARE *
C     * PRIN FOLOSIREA IN CARTELA COMPILE FORTRAN A OPTIUNII CDC, TIPA-*
C     * RIRE NECESARA VERIFICARII MINUTIOASE A VALORILOR INTRODUSE.   *
C     * PENTRU VALORILE TESTULUI SE FOLOSESTE TABELUL ANEXA 3 DIN IN- *
C     * TRODUCERE IN TEORIA STATISTICII DE G.YULE,M.G.KENDALL EDITURA *
C     * STIINTIFICA BUCURESTI 1969. DATELE TABELULUI SE CITESC PE LINII*
C     * SI SE PERFOREAZA CITE 10 VALORI CONSECUTIVE PE O CARTELA.     *
C     ***************************************************************
      DIMENSION HI(30,13)
      READ(105,100)((HI(I,J),J=1,13),I=1,30)
  100 FORMAT(10F6.4)
C     WRITE(108,101)(I,(HI(I,J),J=1,13),I=1,30)
C101 FORMAT(1H1,////1H ,35X,'VALORI   PENTRU  TESTUL   HI  PATRAT '/1H
C    1,35X,36(1H*)///1H ,4X,123(1H*)/1H ,4X,'* *  *',13(8X,1H*)/1H ,4X,'*
C    2  *P*  0.99  *  0.98  *  0.95  *  0.90  *  0.80  *  0.70  *  0.50
C    3 *  0.30  *  0.20  *  0.10  *  0.05  *  0.02  *  0.01  * '
C    4         /1H ,4X,'*NIU**',13(8X,1H*)/1H ,4X,123(1H*)/(1H ,4X,1H*,1
C    5X,I2,2H *,13(F7.4,1X,1H*)))
      RETURN
      END
```

Lista 9.1

Scanned with OKEN Scanner

Lista 9.2

VERIFICAREA IPOTEZEI FLUX DE SOSIRE POISSON

NUMAR DE SOSIRI IN 5 MINUTE		FRECVENTA OBSERVATA		FRECVENTA TEORETICA	
NM(1)=	.0000	FN(1)=	25.0000	FNT(1)=	25.3310
NM(2)=	1.0000	FN(2)=	30.0000	FNT(2)=	29.1306
NM(3)=	2.0000	FN(3)=	17.0000	FNT(3)=	16.7501
NM(4)=	3.0000	FN(4)=	5.0000	FNT(4)=	6.4209
NM(5)=	4.0000	FN(5)=	3.0000	FNT(5)=	2.3675

N= 80.0000 LAMDAT= 1.1500 LAMDA= .2300 HIC= .5174 HI= .5840

IPOTEZA ASUPRA SOSIRILOR ESTE VERIFICATA

VERIFICAREA IPOTEZEI CA SERVIRILE URMEAZA O LEGE EXPONENTIALA

VALORI MEDII A INTERVALELOR DE TIMP		FRECVENTA CUMULATA OBSERVATA		FRECVENTA RELATIVA CUMULATA		FRECVENTA RELATIVA TEORETICA CUMULATA	
DTN1(1)=	1.500	FCN(1)=	33.	FRC(1)=	.412500	FRCT(1)=	.319288
DTN1(2)=	3.500	FCN(2)=	51.	FRC(2)=	.637500	FRCT(2)=	.592387
DTN1(3)=	4.500	FCN(3)=	58.	FRC(3)=	.725000	FRCT(3)=	.684578
DTN1(4)=	5.500	FCN(4)=	64.	FRC(4)=	.800000	FRCT(4)=	.755919
DTN1(5)=	6.500	FCN(5)=	68.	FRC(5)=	.850000	FRCT(5)=	.811124
DTN1(6)=	7.500	FCN(6)=	73.	FRC(6)=	.912500	FRCT(6)=	.853843
DTN1(7)=	8.500	FCN(7)=	75.	FRC(7)=	.937500	FRCT(7)=	.886900
DTN1(8)=	9.500	FCN(8)=	76.	FRC(8)=	.950000	FRCT(8)=	.912481
DTN1(9)=	10.500	FCN(9)=	78.	FRC(9)=	.975000	FRCT(9)=	.932276
DTN1(10)=	11.500	FCN(10)=	79.	FRC(10)=	.987500	FRCT(10)=	.947593
DTN1(11)=	12.500	FCN(11)=	80.	FRC(11)=	1.000000	FRCT(11)=	.959446

D= .093212 LAMDAC= .833716 LAMDAT= 1.230000 MIU= .256410

IPOTEZA SE VERIFICA

Scanned with OKEN Scanner

STUDIUL SISTEMULUI DE ASTEPTARE

LAMDA= ,23000 MIU= ,23041 RO= ,89700

S= 1 P0= ,1030 P= ,8970 N= 2 PN= ,0829 NS= 0,7087 NF= 7,8117 TF=33,9640 TS=37,8640 ROB= ,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 489,48

S= 2 P0= ,3807 P= ,2777 N= 2 PN= ,4023 NS= 1,6475 NF= ,2259 TF= ,9820 TS= 2,9320 ROB= 1,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 69,15

S= 3 P0= ,4047 P= ,0694 N= 2 PN= ,1628 NS= ,9940 NF= ,0296 TF= ,1288 TS= 1,4288 ROB= 2,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 106,99

S= 4 P0= ,4074 P= ,0742 N= 2 PN= ,1039 NS= ,9108 NF= ,0041 TF= ,0178 TS= ,9928 ROB= 3,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 155,40

S= 5 P0= ,4078 P= ,0024 N= 2 PN= ,1640 NS= ,8988 NF= ,0005 TF= ,0023 TS= ,7823 ROB= 4,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 205,18

S= 6 P0= ,4078 P= ,0003 N= 2 PN= ,1641 NS= ,8972 NF= ,0001 TF= ,0003 TS= ,6503 ROB= 5,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 255,15

S= 7 P0= ,4078 P= ,0000 N= 2 PN= ,1641 NS= ,8970 NF= ,0000 TF= ,0000 TS= ,5572 ROB= 6,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 305,15

S= 8 P0= ,4078 P= ,0000 N= 2 PN= ,1641 NS= ,8970 NF= ,0000 TF= ,0000 TS= ,4875 ROB= 7,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 355,15

S= 9 P0= ,4078 P= ,0000 N= 2 PN= ,1641 NS= ,8970 NF= ,0000 TF= ,0000 TS= ,4333 ROB= 8,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 405,15

S=10 P0= ,4078 P= ,0000 N= 2 PN= ,1641 NS= ,8970 NF= ,0000 TF= ,0000 TS= ,3900 ROB= 9,1030
COSTUL MEDIU AL TIMPULUI PIERDUT IN SISTEM ESTE: CS= 455,15

Scanned with OKEN Scanner

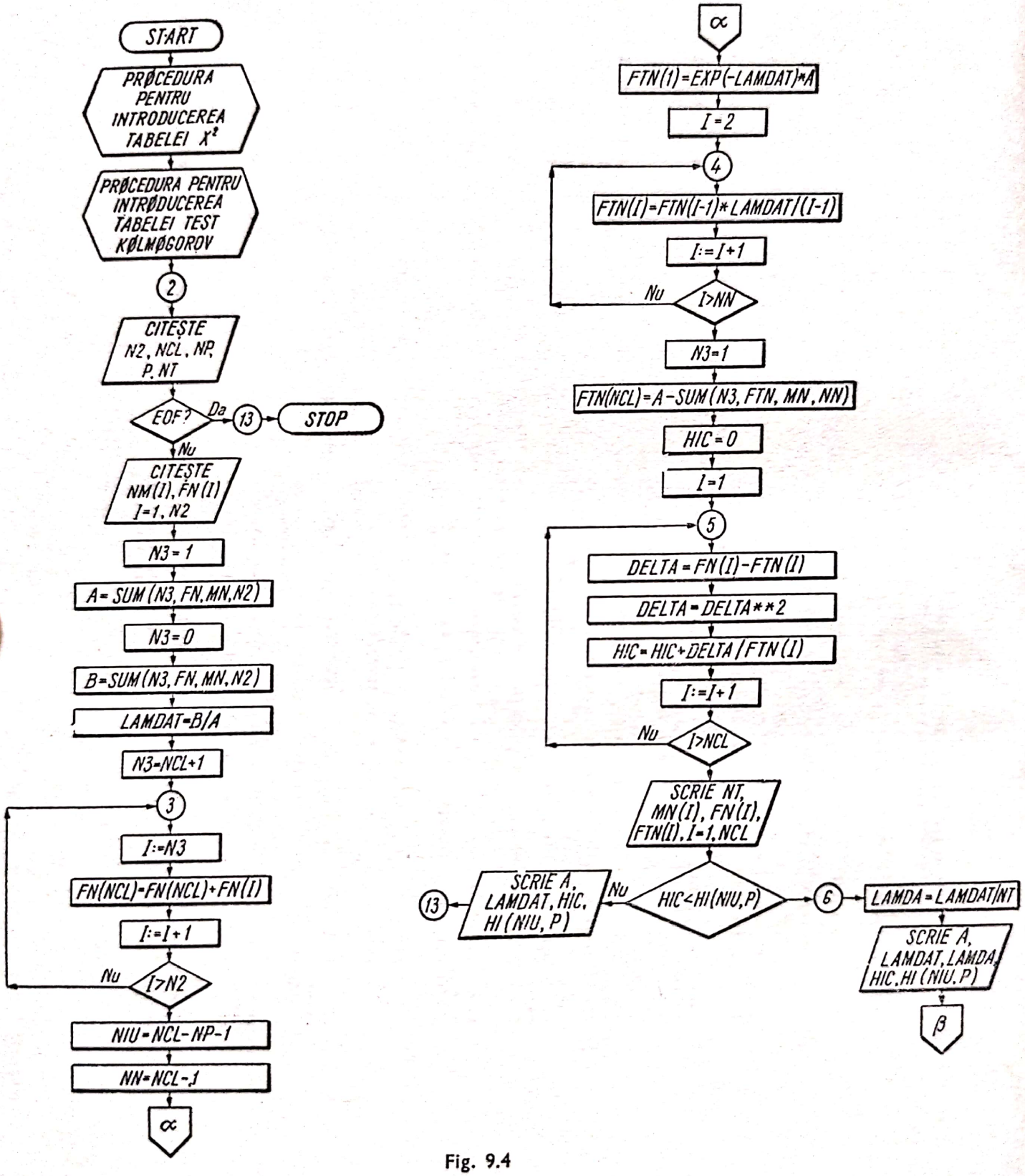
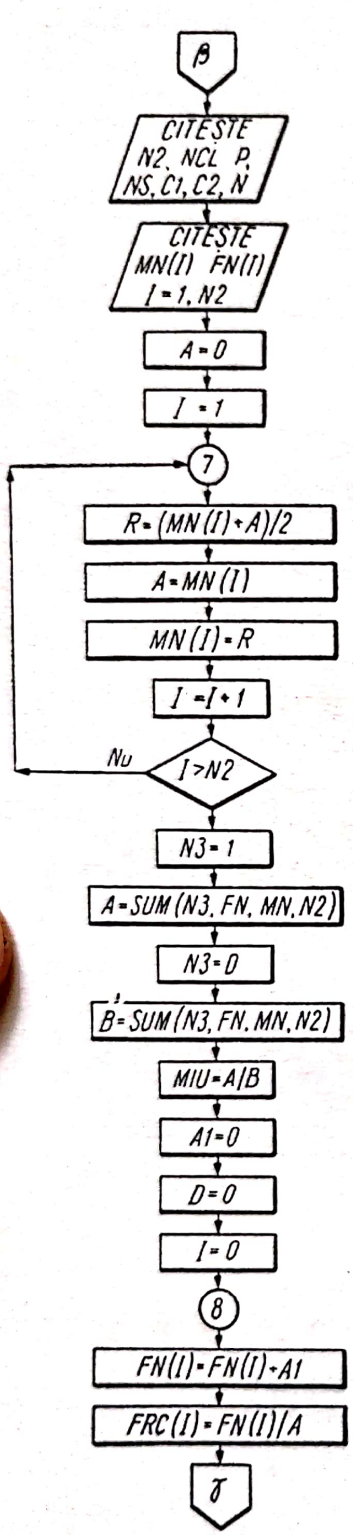
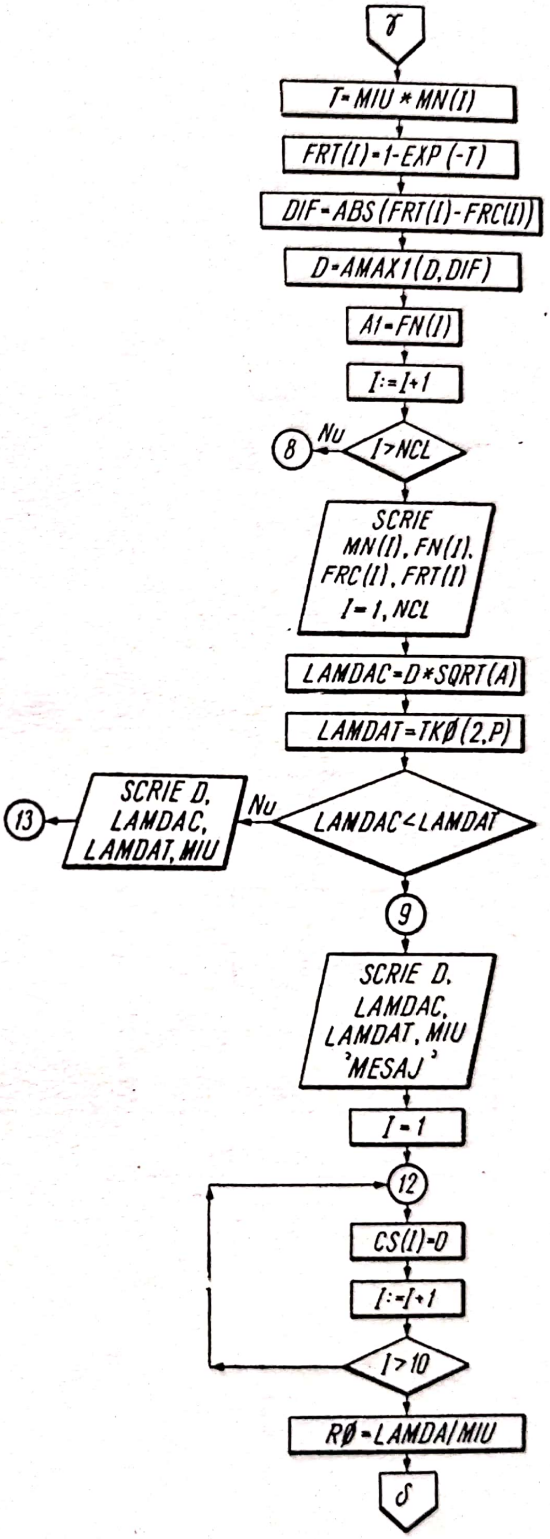


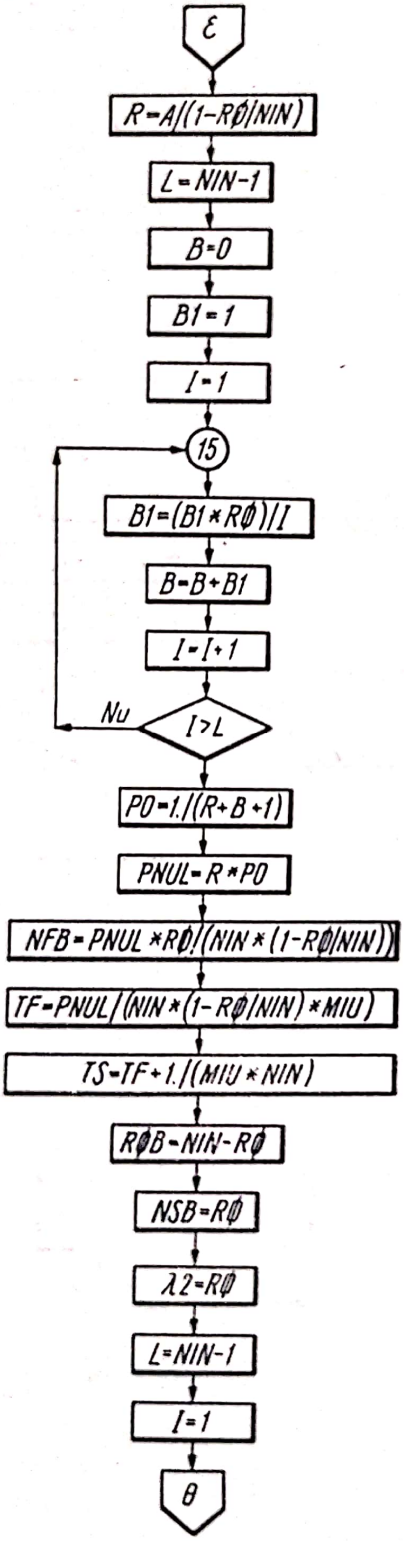
Fig. 9.4

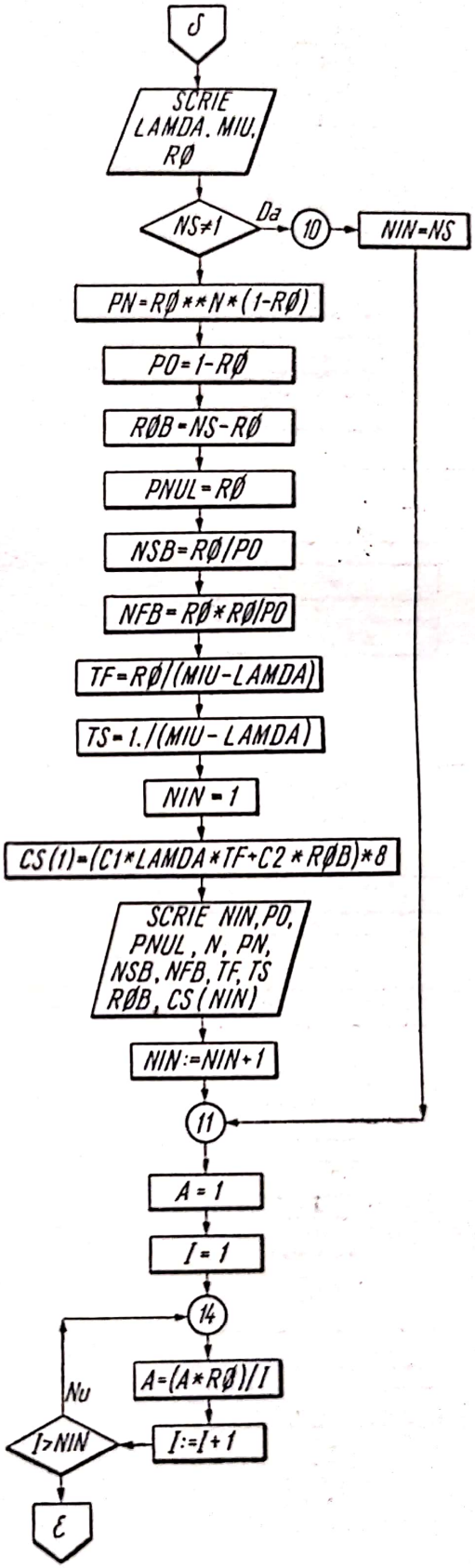
Scanned with OKEN Scanner



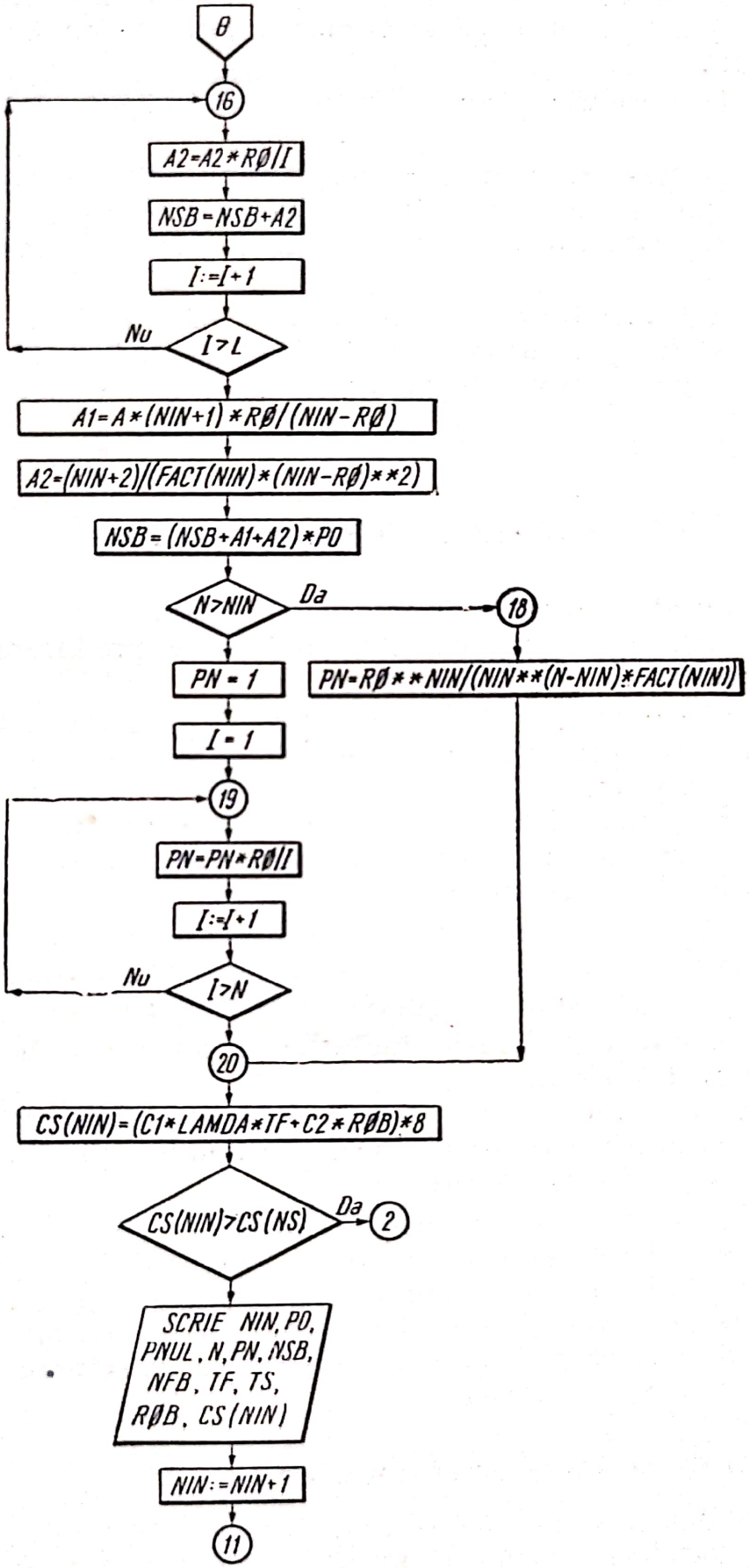


Scanned with OKEN Scanner





Scanned with OKEN Scanner



Scanned with OKEN Scanner

sînt urmate de cartela - EOF. În cazul cînd la o rulare se dorește studiul mai multor sisteme de așteptare, după cartelele cu date corespunzătoare primului sistem se așază cartelele de date corespunzătoare celui de al doilea sistem ș.a.m.d., iar după cartelele de date ale ultimului sistem se va afla - EOF.

Rezultatele obținute pentru exemplul considerat sînt indicate în lista 9.2.

Deosebirile mici ce apar între rezultatele obținute manual și cele obținute cu ajutorul calculatorului electronic se datoresc preciziei mai mari cu care lucrează calculatorul față de precizia cu care s-a lucrat manual.

Pentru a se reduce timpul efectiv al unei rulări, indicăm să se bibliotecheze (catalogheze) programul într-o bibliotecă I.M.T. În felul acesta se elimină timpul necesar compilării și editării la fiecare rulare a programului.

9.2. APLICAȚIILE TEORIEI STOCURILOR

Stocul reprezintă o rezervă de materiale, produse etc., necesare procesului de producție, constituind un „tampon" între producere, consum și reaprovizionare.

Rolul pe care îl are stocul în activitatea de producție este de a înlătura influența neuniformităților provocate de neconcordanța de ritm între producție, consum și aprovizionare. De asemenea, în anumite situații, stocurile de materii și materiale servesc pentru acoperirea unor lipsuri datorite prevederilor greșite sau unor măsuri necorespunzătoare luate de factorii decizionali din întreprinderi.

Din această cauză, cu toate că stocarea implică cheltuieli ce sînt reflectate în prețul de cost, ea are o mare importanță economică, eliminînd fluctuațiile ce pot apare în procesele de producție.

Pe baza acestor considerații, unitățile economice trebuie să-și definească o *politică proprie de stoc*, ținînd cont de condițiile specifice: din interior, în ceea ce privește procesul tehnologic și posibilitățile de depozitare, și de la furnizorii care asigură reaprovizionarea, precum și de prevederile legale cu privire la stocuri.

Din necesitatea de a realiza un program cît mai eficient al stocurilor, s-a dezvoltat în ultima vreme nouă teorie matematică: *teoria stocurilor*, destinată optimizării stocurilor. Aplicarea ei se bazează pe efectuarea unor măsurători și evidențe statistice pentru fiecare produs, pe diferite perioade considerate mai importante, care să permită stabilirea unor modele probabilistice cît mai cuprinzătoare.

Elementele de bază ale teoriei stocurilor

În majoritatea cazurilor ce rezolvă probleme de stoc se urmărește evoluția unei cantități totale Q, de un anumit fel, într-un interval de timp T. Considerînd că această cantitate totală de stoc este o sumă de

Scanned with OKEN Scanner

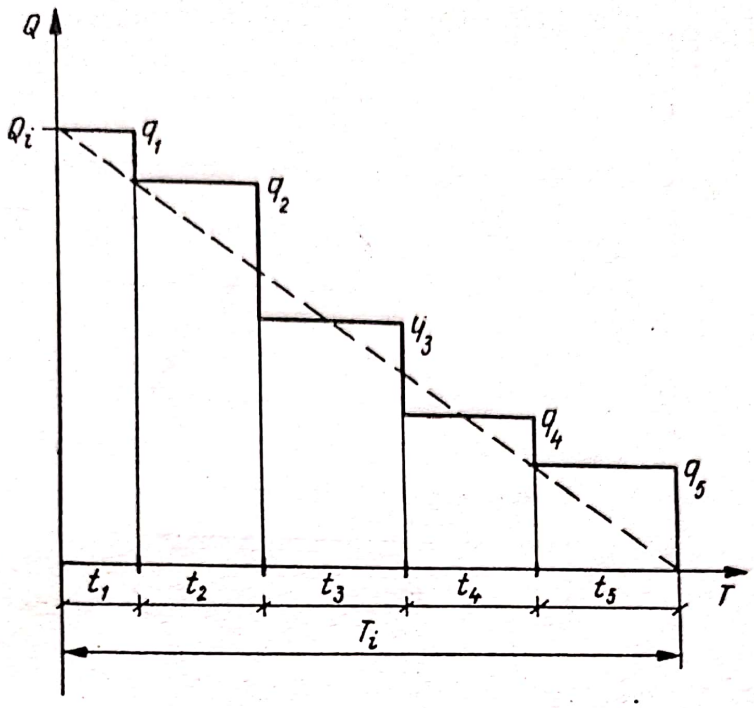


Fig. 9.5

cantităţi discrete : $q_1, q_2, \ldots, q_n$ ce sînt consumate în intervalele de timp : $t_1, t_2, \ldots, t_n$ (fig. 9.5), putem defini *consumul mediu*, C_m, ca fiind raportul dintre cantitatea iniţială a stocului, Q_i, şi perioada totală de consum, T_i, adică :

$$C_m = \frac{Q_i}{T_i}. \tag{9—37}$$

Acest raport reprezintă, în valoare absolută, „înclinarea" *dreptei de consum*.

Admiţînd că din stoc consumul este constant şi că rata de reaprovizionare este instantanee, se poate defini pe o anumită perioadă de timp, T_{ik}, frecvenţa optimă de reaprovizionare.

Cheltuielile ce intervin în cazul organizării stocurilor de producţie pe o perioadă de timp, se pot descompune în :

— cheltuieli de întreţinere a stocului, inclusiv valoarea imobilizată în acesta (K_S) ;

— cheltuieli de reaprovizionare (K_R), reprezentînd valoarea necesară reaprovizionării, independente de cantitatea comandată.

Admiţînd că în intervalul de timp ales, T_{ik}, se face numai o singură reaprovizionare a cantităţii $Q = c \cdot T_{ik}$, cu o cheltuieală de reaprovizionare K_R, se poate obţine o valoare a cheltuielilor de întreţinere : $K_S \cdot \frac{Q}{2} \cdot T_{ik}$, care poate fi destul de ridicată.

Să presupunem că intervalul de timp de reaprovizionare T_{ik} este format din n perioade de timp :

$$T_{ik} = n \cdot t_{ik} \tag{9—38}$$

şi că nivelul stocului este q_m (fig. 9.6.).

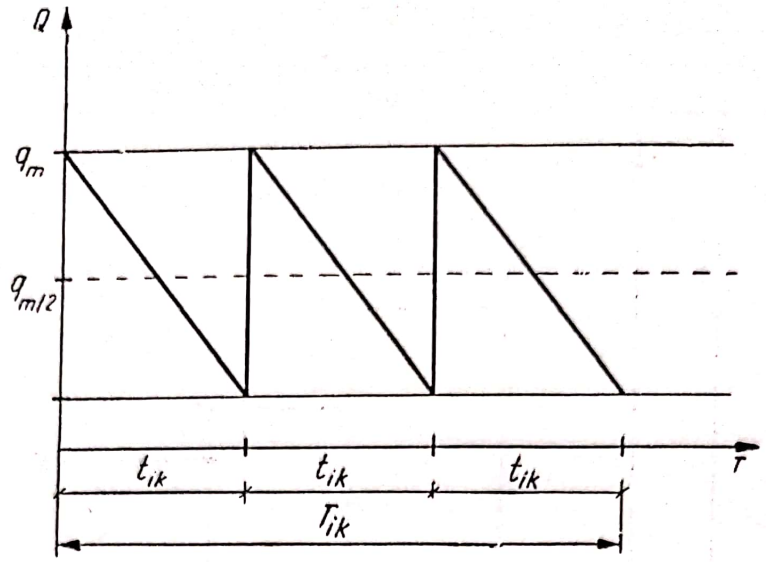


Fig. 9.6

Considerînd stocul mediu (pe fiecare din aceste perioade) : $q_m/2$, atunci cheltuielile de întreţinere pe un interval de timp t_{ik} devin :

$$K_S \cdot \frac{q_m}{2} \cdot t_{ik}. \tag{9—39}$$

Cum am considerat că avem n perioade t_{ik} pe intervalul total de timp T_{ik}, cheltuielile de întreţinere pe această perioadă sînt :

$$n \cdot K_S \cdot \frac{q_m}{2} \cdot t_{ik}, \tag{9—40}$$

iar cheltuielile de reaprovizionare : $n \cdot K_R$.

Avînd aceste două cheltuieli defalcate, se poate obţine valoarea totală a cheltuielilor :

$$E = n \cdot K_S \cdot \frac{q_m}{2} \cdot t_{ik} + n \cdot K_R, \tag{9—41}$$

unde :

$$n = \frac{T_{ik}}{t_{ik}} = \frac{T_{ik} \cdot c}{q_m} = \frac{Q_i}{q_m}.$$

Înlocuind n în expresia (9—41) a lui E, se obţine :

$$E(q_m) = T_{ik} \cdot \frac{q_m}{2} \cdot K_S + \frac{T_{ik} \cdot c}{q_m} \cdot K_R. \tag{9—42}$$

Pentru a determina valoarea optimă ce trebuie avută în stoc pe o anumită perioadă de timp, derivăm funcţia $E(q_m)$, din (9—42), în raport cu q_m şi obţinem :

$$E'(q_m) = \frac{T_{ik}}{2} \cdot K_S - \frac{T_{ik} \cdot c}{q_m^2} \cdot K_R. \tag{9—43}$$

Scanned with OKEN Scanner

Egalînd cu zero valoarea obținută prin derivare obținem:

$$q_m^2 = 2 \cdot C \cdot \frac{K_R}{K_S},$$

sau:

$$q_n^* = \sqrt{2 \cdot C \cdot \frac{K_R}{K_S}} = \sqrt{2 \cdot \frac{Q_i}{T_{ik}} \cdot \frac{K_R}{K_S}}, \qquad (9-44)$$

unde q_m^* reprezintă cantitatea optimă ce trebuie să fie reaprovizionată în intervalul de timp t_{ik}.

Introducînd această valoare în formula cheltuielilor totale, obținem valoarea minimă a acestora:

$$E^* = T_{ik}\sqrt{2 \cdot c \cdot K_R \cdot K_S} = \sqrt{2 \cdot Q_i \cdot T_{ik} \cdot K_R \cdot K_S}. \qquad (9-45)$$

Analizînd formula (9—42), ce reprezintă cheltuielile totale, se poate vedea că este compusă din două funcții: una liniară reprezentînd cheltuielile de întreținere pe o perioadă $\left(T_{ik} \cdot \frac{q_m}{2} \cdot K_S\right)$ și una hiperbolică $\left(\frac{T_{ik} \cdot c}{q_m} \cdot K_R\right)$ ce reprezintă cheltuielile de reaprovizionare. Reprezentînd grafic aceste două funcții, observăm că rezultanta este o asimptotă la dreaptă pentru valorile mari ale stocului și o asimptotă la curbă pentru valorile mici ale stocului (fig. 9.7).

Această reprezentare a funcției cheltuieli este cunoscută sub numele de *modelul lui Wilson*.

Remarcăm că perioada totală T_{ik} este suma perioadelor t_{ik} — timpii necesari reaprovizionărilor cu cantitățile optime q_m^*. Din expresia:

$$n = \frac{T_{ik}}{t_{ik}} = \frac{T_{ik} \cdot c}{q_m},$$

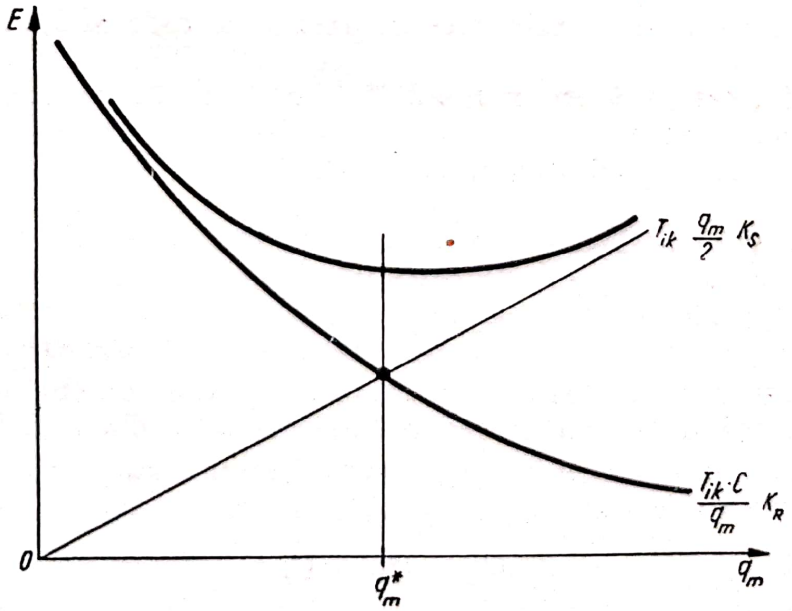


Fig. 9.7

Scanned with OKEN Scanner

se determină numărul optim de reaprovizionări:

$$n^* = \frac{T_{ik} \cdot c}{\sqrt{2 \cdot c \cdot \frac{K_R}{K_S}}}. \qquad (9-46)$$

Analizînd acest număr optim de reaprovizionări, n^*, vedem că pentru a fi mai mare decît unitatea este necesar ca:

$$\frac{K_R}{K_S} < \frac{T_{ik}^2 \cdot c}{2} \text{ sau } K_R < K_S \cdot \frac{T_{ik}^2 \cdot c}{2},$$

adică cheltuielile de reaprovizionare să fie mai mici decît cheltuielile de întreţinere multiplicate cu $T_{ik}^2 c/2$.

Modele probabilistice ale teoriei stocurilor

Modelul descris mai sus, fiind condiţionat de unele ipoteze făcute, nu se apropie de practică. La determinarea cantităţii optime de reaprovizionat, precum şi a numărului optim de reaprovizionări, am admis cheltuielile de întreţinere, K_S, ca fiind proporţionale cu timpul şi cu cantitatea în stoc. Am presupus, de asemenea, cheltuielile de reaprovizionare ca fiind independente faţă de cantitatea de reaprovizionat; mai mult, am considerat consumul ca fiind constant şi rata de reaprovizionare instantanee; în concluzie nu am prevăzut „rupturile" de stoc, ce pot apare pe o anumită perioadă.

În cele ce urmează, depăşim cadrul îngust al modelului descris mai sus, introducînd noi mărimi aleatorii ce pot interveni în problemele de stoc şi anume *cererea* şi *termenul de reaprovizionare* (livrare).

Cererea şi termenul de reaprovizionare pot urmări în evoluţia lor o lege exponenţială de tip Poisson sau o lege normală. Deoarece legile la care ajungem au o expresie analitică, problema poate fi pusă într-o ecuaţie, dar fiind incomodă de rezolvat se utilizează metoda simulării.

Modelul „perioadă cu perioadă". Acest model reconstituie, sau încearcă să reconstituie, la fiecare reaprovizionare un anumit stoc, care poate fi egal cu capacitatea maximă de stocare.

Modelul cu „perioadă fixă". În acest caz regula de reaprovizionare este simplă deoarece datele de reaprovizionare sînt stabilite dinainte, rămînînd să definim cantitatea de reaprovizionare.

Dacă reaprovizionarea este instantanee pot apare trei cazuri:

1° în timpul perioadei T, consumul a fost egal cu stocul maxim, iar cantitatea comandată pentru reaprovizionare este Q_M (fig. 9.8);

2° în timpul perioadei T, consumul a fost mai mic decît stocul maxim Q_M, cantitatea comandată pentru reaprovizionare fiind:

$$Q_C = Q_M - Q_R,$$

unde Q_R este cantitatea de stoc rămasă (fig. 9.9);

Scanned with OKEN Scanner

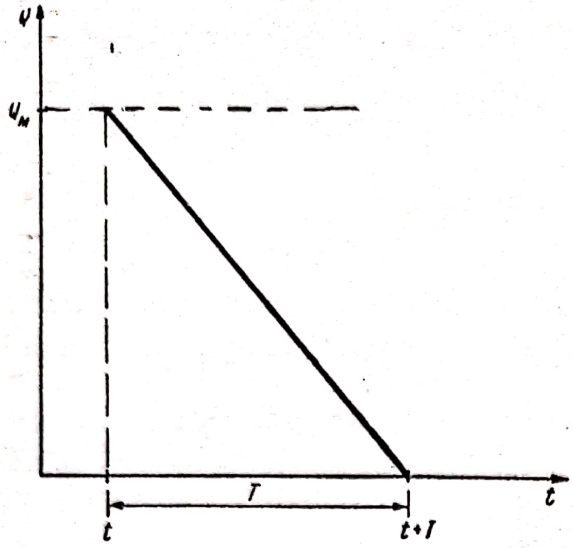


Fig. 9.8

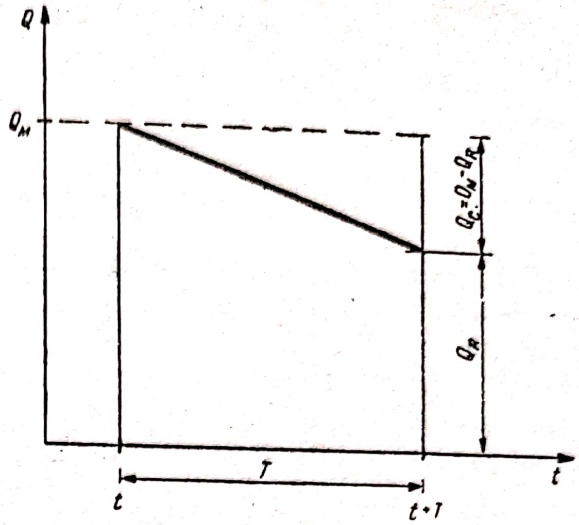


Fig. 9.9

3° în timpul perioadei T, cererea a fost superioară lui Q_M, astfel că stocul scade la valoarea zero înainte de a face o reaprovizionare. Cantitatea de reaprovizionat este Q_M (fig. 9.10).

Avînd în vedere cazurile prezentate mai sus, vom nota: cu Q_S cantitatea de stoc necesară pentru reaprovizionat, cu R rata de reaprovizionare pe o perioadă de timp și cu $P_T(R)$ probabilitatea unei cereri de reaprovizionare pe o perioadă de timp T, cererea fiind presupusă discretă.

Considerăm două perioade de timp T, oarecare; prima perioadă este caracterizată printr-un consum mai mic, iar a doua perioadă printr-un consum mai mare, ceea ce duce la o ruptură de stoc (fig. 9.11).

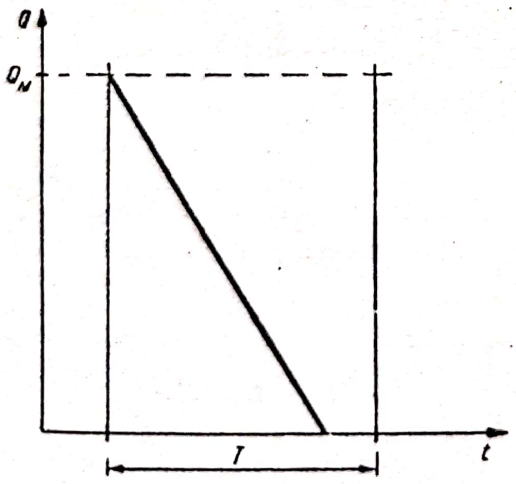


Fig. 9.10

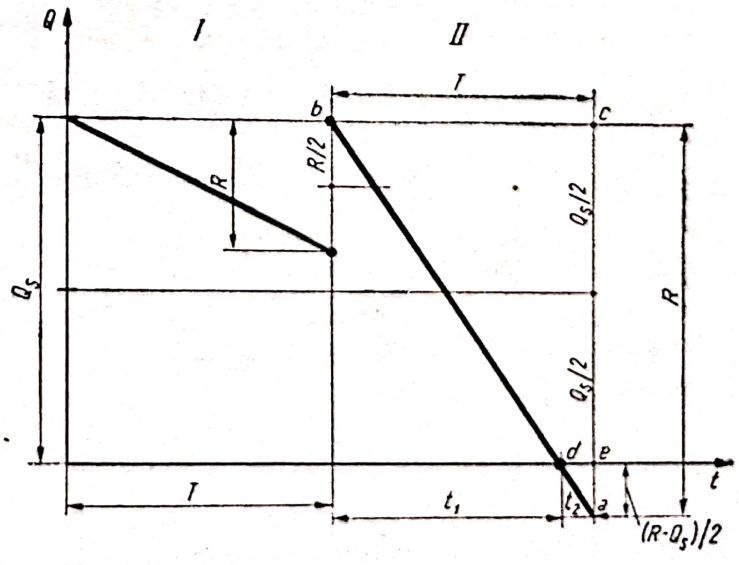


Fig. 9.11

În cazul primei perioade, stocul mediu este: $Q_S - \frac{R}{2}$, iar cheltuielile de întreținere K_S; deci valoarea de întreținere pe perioada T, în acest caz va fi:

$$\left(Q_S - \frac{R}{2}\right) \cdot K_S \cdot T.$$

În cea de a doua perioadă, perioada de timp T este împărțită în perioade de timp t_1, pînă apare ruptura de stoc și t_2 timpul pe perioada rupturii de stoc.

Pe timpul $t_1 = T - t_2$, stocul mediu consumat este $Q_S/2$; pentru a exprima acest timp în funcție de T, Q_S, R vom folosi cele două triunghiuri Δabc și Δadc care sînt asemenea:

$$\frac{R - Q_S}{t_2} = \frac{R}{T} \text{ (din asemănarea triunghiurilor),}$$

$$t_2 = \frac{R - Q_S}{R} \cdot T,$$

$$t_1 = T - t_2 = T - \frac{R - Q_S}{R} \cdot T,$$

$$t_1 = T \cdot \frac{Q_S}{R}.$$

Valoarea de întreținere a stocului pe această perioadă t_1 este deci:

$$\frac{Q_S}{2} \cdot \frac{Q_S}{R} \cdot T \cdot K_S = \frac{Q_S^2}{2R} \cdot T \cdot K_S.$$

Pentru perioada de timp t_2, în cazul apariției rupturii de stoc, așa-numita valoare de penurie este dată de relația:

$$\frac{R - Q_S}{2} \cdot t_2 \cdot K_p;$$

înlocuind valoarea lui t_2, relația de mai sus devine:

$$\frac{R - Q_S}{2} \cdot \frac{R - Q_S}{R} \cdot T \cdot K_p = \frac{(R - Q_S)^2}{2R} \cdot T \cdot K_p,$$

unde K_p reprezintă costul de ruptură de stoc.

În cazul a n perioade T, adăugînd și costul de lansare $n \cdot K_L$ obținem valoarea totală dată de funcția:

$$F = n \sum_{R=0}^{Q_S} \left(Q_S - \frac{R}{2}\right) \cdot K_S \cdot T \cdot P_T(R) + n \sum_{R=Q_S+1}^{\infty} \frac{Q_S^2}{2R} \cdot K_S \cdot T \cdot P_T(R) +$$

$$+ n \sum_{R=Q_S+1}^{\infty} \frac{(R - Q_S)^2}{2R} \cdot T \cdot K_p \cdot P_T(R) + n \cdot K_L \cdot$$

Scanned with OKEN Scanner

Notînd cu θ durata totală de gestiune, $\theta = n \cdot T$, funcţie F devine:

$$F(Q_S,T) = \theta\left[K_S \sum_{R=0}^{Q_S}\left(Q_S - \frac{R}{2}\right)\cdot P_T(R) + K_S \sum_{R=Q_S+1}^{\infty}\frac{Q_S^2}{2R}\cdot P_T(R) + \right.$$

$$\left. + K_p \sum_{R=Q_S+1}^{\infty}\frac{(R-Q_S)^2}{2R}\cdot P_T(R) + \frac{K_L}{T}\right]. \qquad (9-47)$$

În final, alegînd un interval de timp T_0, $2T_0$, $3T_0$, ..., nT_0, şi derivînd în raport cu Q_S, obţinem valoarea optimă a funcţiei $F^*(Q_S^*, T_0)$, $F^*(Q_S^*, 2\ T_0)$, ... care pentru o valoare minimă a stocului constituie valoarea minimă a costurilor totale.

Calculînd un Q_S^* şi $K\ T_0$, acestea vor corespunde unui minimum minimorum, avînd fixat stocul şi perioada de gestiune optim.

Modelul cu „perioadă fixă". Să ne imaginăm un sistem în care stocul iniţial are o anumită valoare, Q_M. Urmărind la intervale de timp, prin calcul, panta dreptei de consum, se poate determina termenul la care cantitatea rămasă în stoc va fi consumată complet.

Analizînd graficul din figura 9.12 constatăm că la timpul t_1 avem un consum care calculat dă o dreaptă cu panta *1*. Dacă, în continuare, consumul urmăreşte această pantă atunci stocul va fi epuizat la timpul $t_1 + d_1$.

Mărind consumul şi calculîndu-l la un timp t_2, obţinem o altă pantă de consum *2*. Urmărind în continuare consumul pe această pantă, stocul va fi epuizat la o dată $t_2 + d_2$, mai mică decît $t_1 + d_1$.

Continuînd acelaşi raţionament se obţine, pe diferite intervale de timp, o curbă frîntă *1 — 2 — 3 —* ... ce reprezintă consumul pe o anumită perioadă de timp. Făcînd media acestor consumuri se obţine un consum mediu cu o pantă medie (fig. 9.13), intervalul de timp $t + d$ reprezentînd timpul mediu la care stocul va fi epuizat. Intervalul de timp d, reprezintă durata în care se face o reaprovizionare cu cantitatea Q_M.

Această cerere de reaprovizionare se face în momentul în care cantitatea rămasă în stoc acoperă consumul corespunzător termenului de reaprovizionare d.

În realitate, datorită unor condiţii externe pot apare diferite cazuri.

În figura 9.14 se arată situaţia în care la timpul t_1 se lansează cererea de reaprovizionare, stocul rămas acoperind consumul constant pe durata de timp d. Mărindu-se consumul datorită unor cereri suplimentare se va produce o ruptură de stoc înainte de a se face reaprovizionarea.

Dacă la timpul t_2 (fig. 9.14) se lansează cererea de reaprovizionare, deoarece rata de consum este mai mică decît cea calculată, atunci la sfîrşitul perioadei d va mai exista o cantitate în stoc, astfel că după reaprovizionare stocul va fi mai mare decît stocul iniţial: $Q'_M > Q_M$.

Pentru evitarea acestor două cazuri extreme, mai ales ruptura de stoc, precum şi pentru determinarea punctului de comandă a reaprovizionării se introduce un nivel de stoc, denumit *stocul de alertă*: Q_A (fig. 9.15).

Scanned with OKEN Scanner

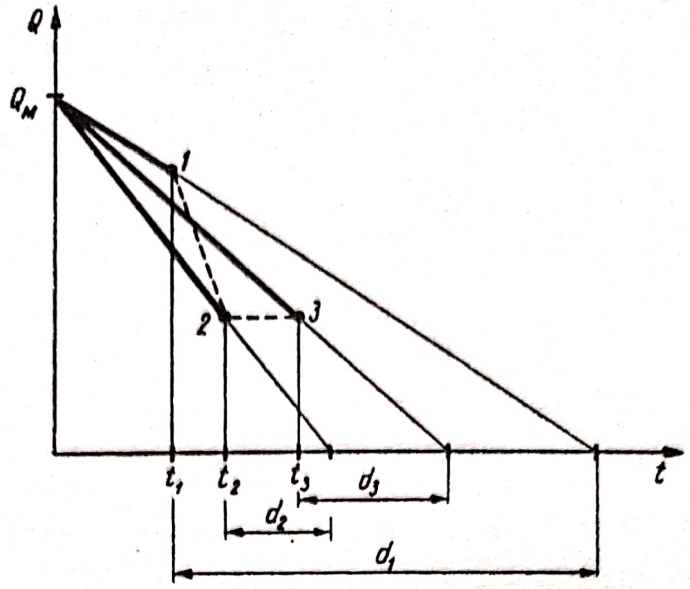


Fig. 9.12

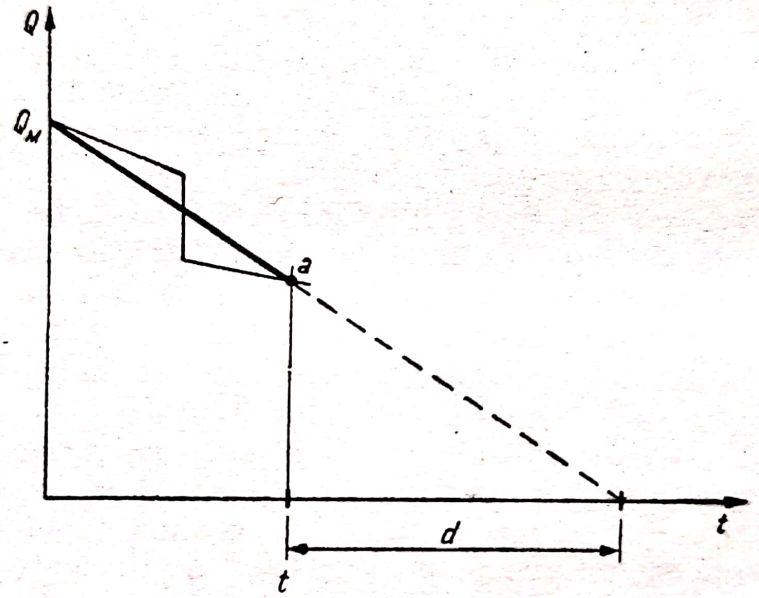


Fig. 9.13

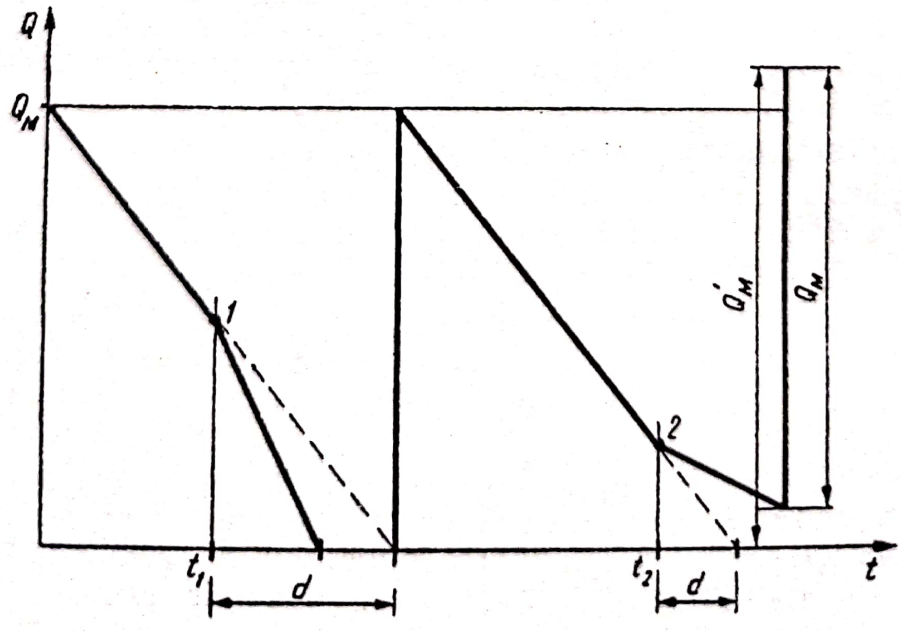


Fig. 9.14

Scanned with OKEN Scanner

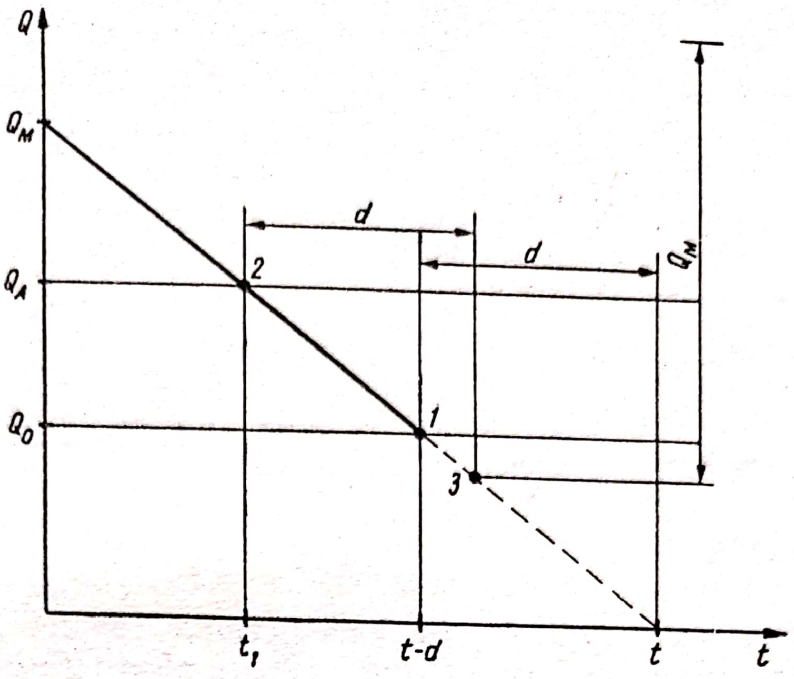


Fig. 9.15

În modelul descris mai sus, la timpul $t-d$ (fig. 9.15) se cerea reaprovizionarea stocului, stocul Q_0 (corespunzător punctului *1*) putînd fi denumit şi stoc de alertă limită, el fiind în acest caz egal cu consumul mediu pe perioada de timp d.

Pentru a diminua riscul de a nu avea ruptură de stoc, se ridică pragul stocului de alertă ($Q_A > Q_0$), ceea ce va provoca o cerere de reaprovizionare la timpul $t_1 < t-d$ (punctul *2* în figura 9.15).

În acest caz rămîne de determinat cu ce cantitate urmează a se face reaprovizionarea pentru a nu avea un excedent de stoc. Această cantitate poate fi apreciată, conform previziunilor de moment, pentru recompletarea stocului la timpul $t_1 + d$ după relaţia:

$$Q = Q_M - (Q_A - Q_0) = Q_M - Q_A + Q_0.$$

Supraveghind constant dreapta de consum şi făcînd $Q_0 = 0$, se poate determina timpul t_1 la care stocul va atinge stocul Q_A (fig. 9.16).

Comanda de reaprovizionare, în acest caz, se va face la timpul t_1-d, avînd o valoare de stoc $Q_M - Q_A$, ce reprezintă necesarul de recompletare a stocului.

Deoarece fixarea stocului de alertă: Q_A, precum şi cantitatea fixă: $Q_M - Q_A$ sînt greu de stabilit, se introduce *stocul de securitate* definit prin două nivele:

Q_c — *stocul de comandă*, care determină momentul comenzii prin intersecţia cu dreapta de consum;

Q_S — stocul la care se ajunge prin modificări, aduse în funcţie de politica de livrare; de exemplu, livrări la anumiţi beneficiari (fig. 9.17).

Această metodă aduce siguranţă mai mare în ceea ce priveşte ruptura de stoc, precum şi un mod comod de reaprovizionare.

Scanned with OKEN Scanner

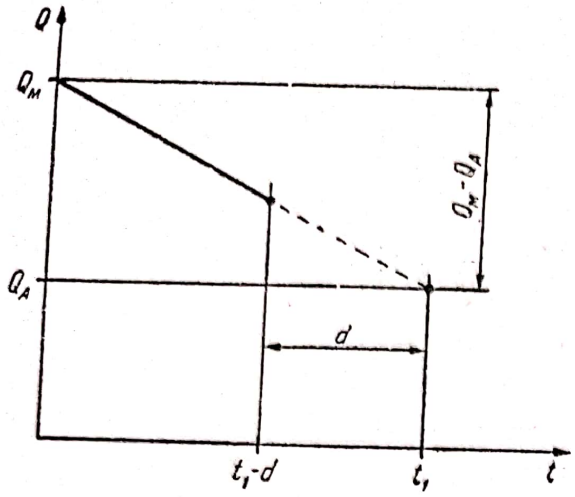


Fig. 9.16

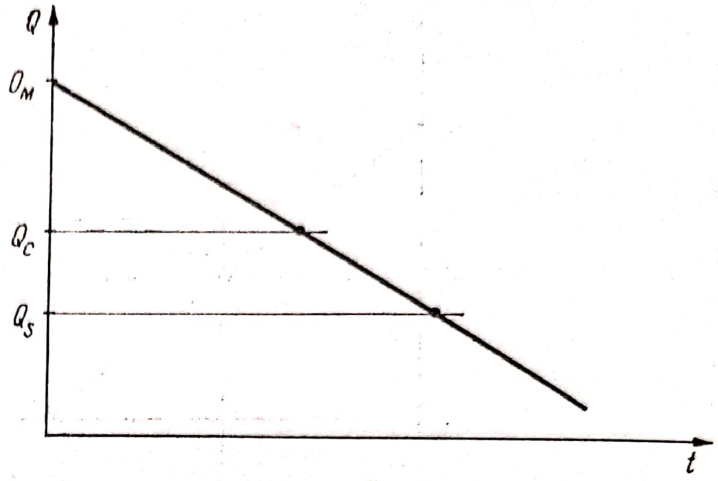


Fig. 9.17

Vom încheia acest capitol prin două exemple concrete de aplicare a teoriei stocurilor, pe baza unor calcule analitice efectuate cu ajutorul sistemelor de calcul automat.

Exemple

Ex. 9—1. Exemplul de mai jos reprezintă un *calcul automat al stocului de siguranță*. Considerînd cazul unei magazii de piese de schimb, interesant este să se cunoască stocul optim de siguranță pentru ca probabilitatea să apară o ruptură de stoc să fie foarte mică, avînd în vedere faptul evident că stocînd o cantitate mare de piese prețul va fi destul de ridicat.

Observînd evoluția cererii pentru o singură piesă s-a stabilit, de exemplu, următoarea tabelă cu valori ale probabilităților de cerere a diferitelor cantități din aceeași piesă :

Cererea piesei R (buc)	0	1	2	3	4	5	6	7	8
Probabilitatea $P(R)$	0,03	0,08	0,15	0,30	0,15	0,15	0,05	0,05	0,04

Avînd în vedere cele spuse anterior, pot exista două cazuri limită : unul cînd stocul este superior cererii R (acesta nefiind interesant de studiat) și al doilea cînd cererea este superioară stocului, apărînd ruptura de stoc. Folosind formulele stabilite anterior (9—47), valoarea costului global se ridică la :

$$F(Q_S) = K_S \cdot \sum_{R=0}^{Q_S} \left(Q_S - \frac{R}{2}\right) \cdot P(R) + K_S \sum_{R=Q_S+1}^{\infty} \frac{Q_S^2}{2R} \cdot P(R) +$$

$$+ K_p \sum_{R=Q_S+1}^{\infty} \frac{(R - Q_S)}{2R} \cdot P(R), \qquad (9—48)$$

Scanned with OKEN Scanner

unde, spre exemplificare, considerăm: prețul de stocaj $K_S = 40$ lei și prețul de ruptură de stoc $K_p = 200$ lei.

Notînd: $RO = \frac{K_p}{K_S + K_p}$, stocul optim de siguranță se va determina astfel ca: $E(Q_{S_0} - 1) < RO < E(Q_{S_0})$ să fie îndeplinită. În continuare prezentăm modul cum se determină valoarea acestei funcții $E(Q_{S_0})$.

Utilizînd expresia (9—48) obținem:

$$F(Q_S + 1) = K_S \sum_{R=0}^{Q_{S+1}} \left(Q_S + 1 - \frac{R}{2}\right) p(R) + K_S \sum_{R=Q_S+2}^{\infty} \frac{1}{2} \cdot \frac{(Q_S + 1)^2}{R} p(R) +$$

$$+ K_p \sum_{R=Q_S+}^{\infty} \frac{1}{2} \frac{(R - Q_S - 1)^2}{R} p(R).$$

Se poate scrie:

$$K_S \sum_{R=0}^{Q_{S+1}} \left(Q_S + 1 - \frac{R}{2}\right) p(R) = K_S \sum_{R=0}^{Q_S} \left(Q_S + 1 - \frac{R}{2}\right) p(R) +$$

$$+ K_S \left(Q_S + 1 - \frac{Q_S + 1}{2}\right) p(Q_S + 1) = K_S \sum_{R=0}^{Q_S} \left(Q_S - \frac{R}{2}\right) p(R) +$$

$$+ K_S \sum_{R=0}^{Q_S} p(R) + K_S \frac{Q_S + 1}{2} p(Q_S + 1).$$

În același mod se transformă și ceilalți termeni ai funcției $F(Q_S + 1)$, adică:

$$K_S \sum_{R=Q_S+2}^{\infty} \frac{(Q_S + 1)^2}{2R} p(R) = K_S \sum_{R=Q_S+1}^{\infty} \frac{Q_S^2}{2R} p(R) + K_S \sum_{R=Q_S+1}^{\infty} Q_S \cdot \frac{p(R)}{R} +$$

$$+ K_S \sum_{R=Q_S+1}^{\infty} \frac{p(R)}{2R} - K_S \frac{(Q_S + 1)}{2} p(Q_S + 1)$$

și

$$K_p \sum_{R=Q_S+2}^{\infty} \frac{(R - Q_S - 1)^2}{2R} = K_p \sum_{R=Q_S+1}^{\infty} \frac{(R - Q_S)^2}{2R} p(R) - K_p \sum_{R=Q_S+1}^{\infty} p(R) +$$

$$+ K_p \sum_{R=Q_S+1}^{\infty} Q_S \frac{p(R)}{R} + K_p \cdot \frac{1}{2} \sum_{R=Q_S+1}^{\infty} \frac{p(R)}{R}.$$

Observăm că:

$$F(Q_S + 1) = F(Q_S) + K_S \sum_{R=0}^{Q_S} p(R) + K_S \frac{Q_S + 1}{2} p(Q_S + 1) +$$

Scanned with OKEN Scanner

$$+ K_S \sum_{R=Q_S+1}^{\infty} Q_S \frac{p(R)}{R} + K_S \sum_{R=Q_S+1}^{\infty} \frac{p(R)}{2R} - K_S \frac{(Q_S+1)}{2} p(Q_S+1) -$$

$$- K_p \sum_{R=Q_S+1}^{\infty} p(R) + K_p \sum_{R=Q_S+1}^{\infty} Q_S \frac{p(R)}{R} + K_p \cdot \frac{1}{2} \sum_{R=Q_S+1}^{\infty} \frac{p(R)}{R}.$$

Grupînd termenii rezultă:

$$F(Q_S+1) = F(Q_S) + (K_S + K_p)\left[p(R \leqslant Q_S) + \left(Q_0 + \frac{1}{2}\right) \sum_{R=Q_S+1}^{\infty} \frac{p(R)}{R}\right] - K_p,$$

unde notăm expresia dintre parantezele drepte cu $E(Q_S)$, adică:

$$E(Q_S) = p(R \leqslant Q_S) + (Q_S+1) \sum_{R=Q_S+1}^{\infty} \frac{p(R)}{R}$$

Dacă cererea este o variabilă aleatorie continuă se poate scrie:

$$F(Q_S) = K_S \int_0^{Q_S} \left(Q_S - \frac{R}{2}\right) V(R)\, dR + K_S \int_{Q_S}^{\infty} \frac{Q_S^2}{2R} V(R)\, dR +$$

$$+ K_p \int_{Q_S}^{\infty} \frac{(R-Q_S)^2}{2R} V(R)\, dR.$$

Determinăm expresia derivatei acestei funcții în raport cu Q_S; vom obține:

$$\frac{\partial F}{\partial Q_S} = K_S \int_0^{Q_S} V(R)\, dR + K_S \int_{Q_S}^{\infty} \frac{Q_S}{R} V(R)\, dR + K_p \int_{Q_S}^{\infty} \frac{Q_S - R}{R} V(R)\, dR$$

care, pentru a avea valoarea optimă Q_{S0}^*, trebuie $\frac{\partial F}{\partial Q_S} = 0$. Rezultă:

$$(K_p + K_S) \cdot \left(\int_{Q_{S0}}^{\infty} \frac{Q_{S0}}{R} V(R)\, dR + \int_0^{Q_{S0}} V(R)\, dR\right) = K_p \int_0^{\infty} V(R)\, dR,$$

$$(K_p + K_S) \cdot \left(\int_{Q_{S0}}^{\infty} \frac{Q_{S0}}{R} V(R)\, dR + \int_0^{Q_{S0}} V(R)\, dR\right) = K_p,$$

$$Q_{S0} \int_{Q_{S0}}^{\infty} \frac{V(R)}{R}\, dR + \int_0^{Q_{S0}} V(R)\, dR = \frac{K_p}{K_p + K_S} = RO$$

sau:

$$RO = F(Q_S) + Q_{S0} \int_{Q_S}^{\infty} \frac{V(R)}{R}\, dR.$$

Scanned with OKEN Scanner

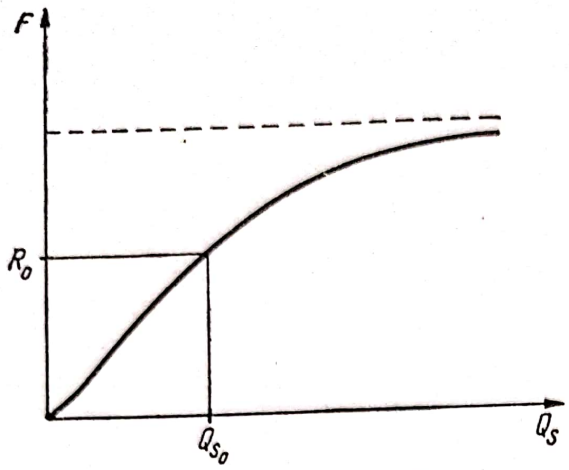


Fig. 9.18

În concluzie, calculînd raportul RO și determinînd funcțiile $E(Q_{S0}-1)$ și $E(Q_{S0})$ obținem valoarea optimă a stocului de siguranță $Q^*_{S_0}$ (fig. 9.18).

Calculul automat al stocului optim de siguranță, prin utilizarea unui calculator electronic, se poate realiza conform schemei de calcul mai generală din figura 9.19, care — pentru a fi în concordanță cu denumirile de variabile din programul FORTRAN care o descrie, utilizează notațiile C_S și C_P pentru costurile de stocare și de ruptură (în loc de K_S și — respectiv — K_p, utilizate pînă acum).

Programul FORTRAN aferent este prezentat în „out-print"-ul din lista 9.3, iar un exemplu de rezultate obținute cu acest program este redat în lista 9.4. Constatăm că pentru valorile date în *Ex. 9—1* s-a obținut $RO = 0{,}8333$; deoarece el este cuprins între valorile 0,7299 și 0,8679 se deduce că $Q^*_{S_0} \simeq 2$, adică stocul optim de siguranță pentru cazul din *Ex. 9—1*.

Ex. 9—2. În încheiere, vom prezenta un exemplu de *simulare a gestiunii stocurilor*. Vom căuta ca pe baza unor observații statistice făcute la o magazie de materiale, să se simuleze o gestiune de stoc în așa fel încît costul global să fie cît mai mic, iar probabilitatea de a pare o ruptură de stoc să fie cît mai mică.

Într-o gestiune de stoc, cererea unui articol este urmată fie de livrarea sa, fie de apariția rupturii de stoc. Livrarea articolului atrage imediat după ea punerea la zi a stocului și reaprovizionarea prin comandă.

În acest sens, se poate face o organigramă care să sugereze succesiunea operațiilor elementare în urmărirea unei gestiune de stoc (fig. 9.20).

Studiindu-se caietul de gestiune al magaziei s-a observat că cererea este aleatoare și că probabilitatea p a unei cereri zilnice de n articole este, de exemplu, cea din următoarea tabelă:

Numărul de articole (n)	0	1	2	3	4	5	6	7	8	9	10
Probabilitatea $p(n)$ în%	3	5	10	20	30	10	8	5	4	3	2

Utilizînd și datele cu privire la frecvența de observație (v. lista 9.9), simularea se poate face conform schemei de calcul din figura 9.21, pentru care am întocmit programul FORTRAN prezentat în lista 9.5 (ce conține, prin linii — comentarii, și toate indicațiile cu privire la utilizarea lui). Rezultatele aplicării acestui program, pentru cazul considerat prin datele din tabela de mai jos, cu $p(n)$, sînt redate în listele 9.6, ..., 9.11.

Scanned with OKEN Scanner

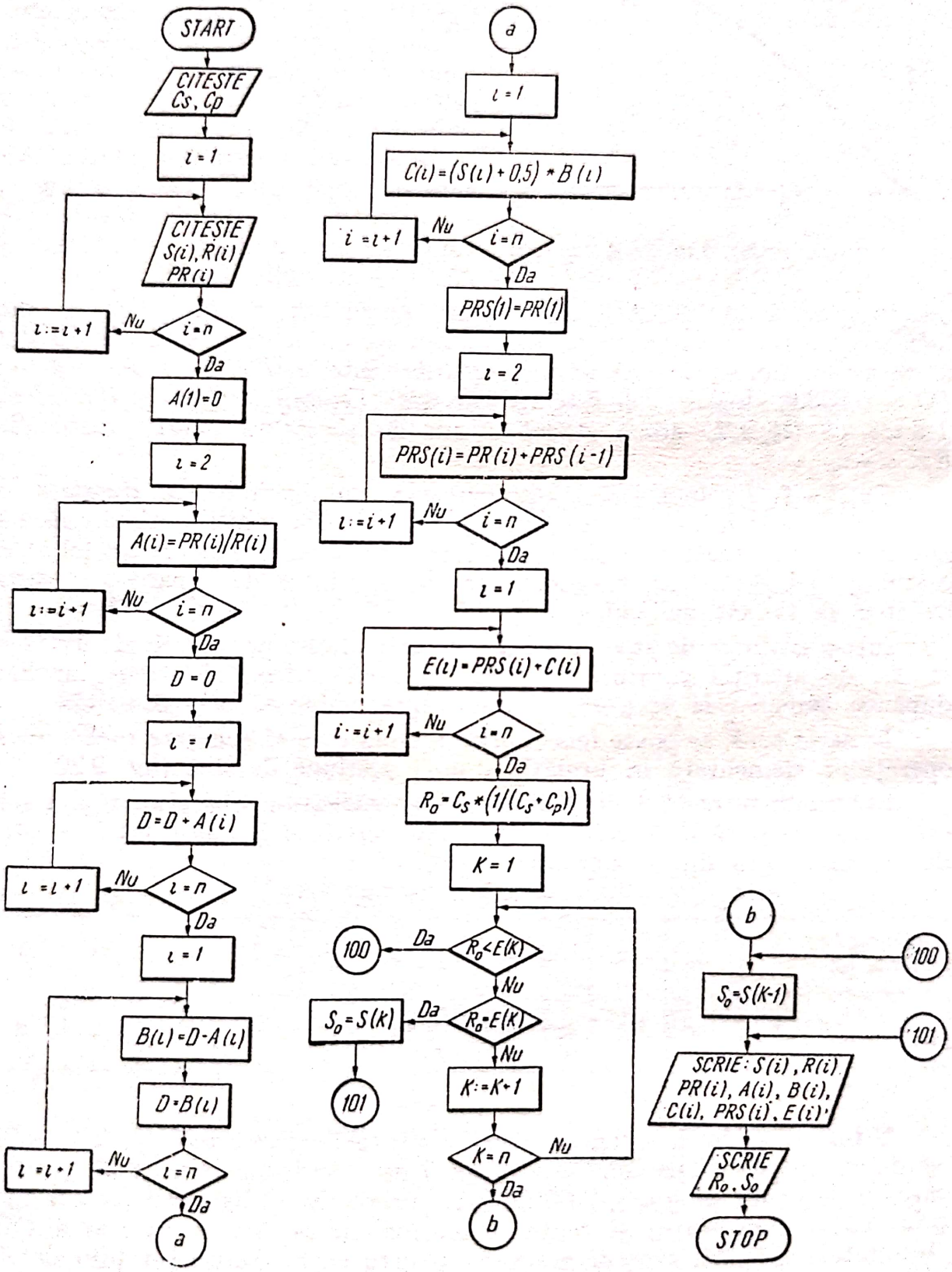


Fig. 9.19

Scanned with OKEN Scanner

```
C      LISTA 9.3
C      ************************************
C      * PROGRAM FORTRAN PENTRU CALCULUL  *
C      * STOCULUI OPTIM DE SIGURANTA      *
C      ************************************
C      PROGRAMUL A FOST RULAT PENTRU DATELE
C      DIN EX.9-1,IN CONFORMITATE CU SCHEMA
C      DE CALCUL DIN FIG.9.19.

C      DATELE INTRODUSE IN PROGRAM SINT:
C       -COSTUL DE STOCAJ
C       -COSTUL DE RUPTURA
C       -NIVELUL STOCULUI
C       -CEREREA
C      -PROBABILITATEA CERERII
C      MACHETELE DE PERFORARE PENTRU ACESTE
C      DATE SINT:
C      -MACHETA 1: CS,CP INCEPIND DIN
C                  COLOANA 1 2I3
C
C      -MACHETA 2: S(I),R(I),PR(I) INCEPIND
C                  DIN COLOANA 1 2I3,F3.2
C
C      PROGRAMUL DETERMINA STOCUL OPTIM PE BAZA
C      OBSERVATIILOR PRIVIND CEREREA UNUI LOT DE
C      PIESE SI A PROBABILITATII APARITIEI CERERII
C      PROGRAMUL POATE FI FOLOSIT PENTRU ORICE
C      STOC CERERE CU CONDITIA CA SUMA(PR(I))=1

C      SEMNIFICATIA UNOR NUME DE VARIABILE UTILIZATE
C      IN ACEST PROGRAM ESTE URMATOAREA(SEMNUL= AVIND
C      SEMNIFICATIA DE '' REPREZINTA''):
C      CS=COSTUL DE STOCAJ PE PIESA
C      CP=COSTUL DE RUPTURA PE PIESA
C      S=STOCUL
C      R=CEREREA
C      PR=PROBABILITATEA CERERII
C      B=SUMA RAPORTULUI PROBABILITATE/CERERE
C      A=RAPORTUL PROBABILITATE/CERERE
C      C=FUNCTIA STOCULUI OPTIM
C      PRS=PROBABILITATEA SITUATIILOR R<S
C      E=FUNCTIA STOCULUI DE SIGURANTA
C      RD=RAPORTUL COSTURILOR
C      SO=STOCUL DE SIGURANTA OPRIM
       DIMENSION PR(100),A(100),B(100),C(100),PRS(100),E(100)
       INTEGER S(100),R(100)
       READ(105,18)CS,CP
    18 FORMAT(2I3)
       N=9
       DO 1 I=1,N
     1 READ(105,2)S(I),R(I),PR(I)
     2 FORMAT(2I3,F3.2)
       A(1)=0
       DO 3 I=2,N
     3 A(I)=PR(I)/R(I)
       D=0
       DO 4 I=1,N
     4 D=D+A(I)
       DO 5 I=1,N
       B(I)=D-A(I)
     5 D=B(I)
       DO 6 I=1,N
```


Scanned with OKEN Scanner

```
      6 C(I)=(S(I)+0.5)*B(I)
        PRS(1)=PR(1)
        DO 7 I=2.N
      7 PRS(I)=PR(I)+PRS(I-1)
        DO 8 I=1,N
      8 E(I)=PRS(I)+C(I)
        RO=CP*(1/(CS+CP))
        K=1
    103 IF(RO.LT.E(K)) GO TO 100
        IF(RO.EQ.E(K))GO TO 101
        K=K+1
        IF(K.EQ.N)GO TO 102
        GO TO 103
    100 SO=S(K-1)
              GO TO 104
    101 SO=S(K)
    104 WRITE(108,10)
     10 FORMAT(' ',18X,'D E T E R M I N A R E A   S T O C U L U I   O P T
       *I M  SO',///)
        WRITE(108.20)CS,CP
     20 FORMAT(' ',10X,'COSTUL DE STOCAJ =',I3,1X,'LEI'//' ',10X,'COSTUL D
       *E RUPTURA =',I3,1X,'LEI'///)
        WRITE(108,11)
     11 FORMAT(' ',6X,78(1H*)/' ',6X,1H*,5X,1H*,5X,1H*,6X,1H*,8X,1H*,11X,1
       *H*,18X,1H*,8Y,1H*,8X,1H*/' ',6X,1H*,'  S   *  R   * P(R)  * P(R)/R *
       *S(P(R)/R) *  (S+0.5)S(P(R)/R)  * P(R,S)  *   E(S)    *'/' ',6X,'*
       *     *        *          *                  *                     *
       *     *'/' ',6X,78(1H*))
        DO 12 I=1,N
     12 WRITE(108,14)S(I),R(I),PR(I),A(I),B(I),C(I),PRS(I),E(I)
     14 FORMAT(' ',6X,1H*,1X,I3,1X,1H*,1X,I3,1X,1H*,1X,F4.2,1X,1H*,1X,F6.4
       *,1X,1H*,2X,F6.4,3X,1H*,5X,F7.5,6X,1H*,2X,F4.2,2X,1H*,1X,F6.4,1X,1H
       **)
        WRITE(108,15)RO,SO
     15 FORMAT(' ',6X,78(1H*)////' ',12X,'S T O C U L  O P T I M :  RO=',F6
       *.4,3X,'SO=',F6.4)
        GO TO 106
    102 WRITE(108,17)
     17 FORMAT('1',5X,'SOLUTIE IMPOSIBILA')
    106 STOP
        END
```

Lista 9.3

```
RUN
STARTED
```

D E T E R M I N A R E A   S T O C U L U I   O P T I M   SO

COSTUL DE STOCAJ = 40 LEI

COSTUL DE RUPTURA =200 LEI

S	R	p(R)	p(R)/R	S(p(R)/R)	(S+0,5)S(P(R)/R)	P(R,S)	E(S)
0	0	.03	.0000	.3430	.17149	.03	.2015
1	1	.09	.0800	.2630	.39446	.11	.5045
2	2	.15	.0750	.1880	.46994	.26	.7299
3	3	.30	.1000	.0880	.30792	.56	.8679
4	4	.15	.0375	.0505	.22714	.71	.9371
5	5	.15	.0300	.0205	.11262	.86	.9726
6	6	.05	.0083	.0121	.07893	.91	.9889
7	7	.05	.0071	.0050	.03750	.96	.9975
8	8	.04	.0050	-.0000	-.00000	1.00	1.0000

S T O C U L   O P T I M :   R O= .8333   SO=2.0009

STOP

Lista 9.4

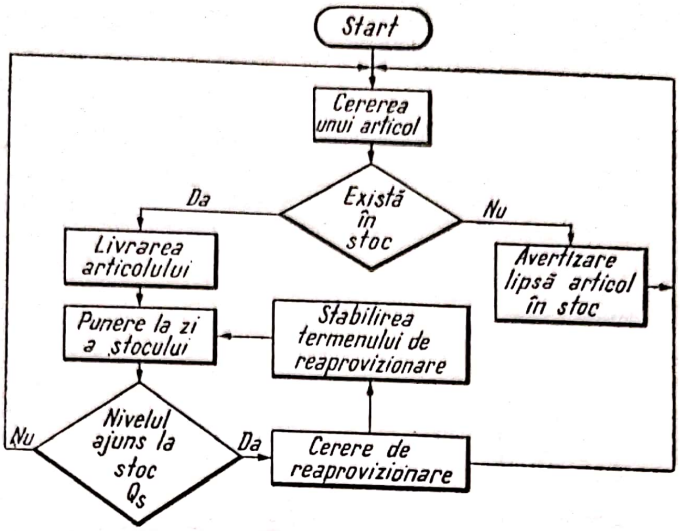


Fig. 9.20

În lista 9.9 (intitulată „Calculul mediei distribuției") este dată frecvența observată (în comparație cu cea teoretică) precum și calculul mediei frecvenței, care (cu scrierea din lista 9.6) este: $MEDIA\ FRECVENȚEI = \frac{1}{100}\sum (FRECVENȚA\ OBSERVATA)*CIFRE$

Metoda de simulare constă în a comanda în ultima zi lucrătoare a fiecărei săptămîni un număr fix de articole egal cu *consumul mediu* („media frecvenței ori numărul de zile lucrătoare").

Etapele simulării gestiunii de stoc sînt următoarele:

1° *Generarea eșantioanelor artificiale ale cererii.* Se bazează pe generarea numerelor echiprobabile printr-o metodă oarecare, în cazul de față metoda „mid-square", care constă în a lua un număr „la hazard" de $2n$ cifre, a-l ridica la pătrat, iar apoi a extrage $2n$ cifre centrale care dau pe n_1, după care se face din nou aceeași operație (v. lista 9.6). De exemplu:

$$n_0 = 7396;\ n_0^2 = \overset{54\,70\,08\,16}{\underset{n_1}{\overline{\qquad}}}$$

$$n_1 = 7008;\ n_1^2 = \overset{49\,11\,20\,64}{\underset{n_2}{\overline{\qquad}}}$$

. . . . . .

Șirul de numere rezultat: 73967008 ... poate fi considerat un șir de numere echiprobabile.

Printr-un test χ^2 aplicat, de exemplu, primelor 100 de cifre aflăm dacă putem admite șirul de numere echiprobabile în raport cu cît dorim să fie mai aproape de realitate simularea. Frecvența teoretică se calculează cu relația:

$$P_{n(t)} = \frac{\lambda(t)^n}{n!}\,e^{-\lambda t},$$

unde $\lambda t = \sum_{0}^{nr.\ de\ cifre} \frac{n \cdot frecvența\ observată}{nr.\ total\ de\ cazuri}$.

Scanned with OKEN Scanner

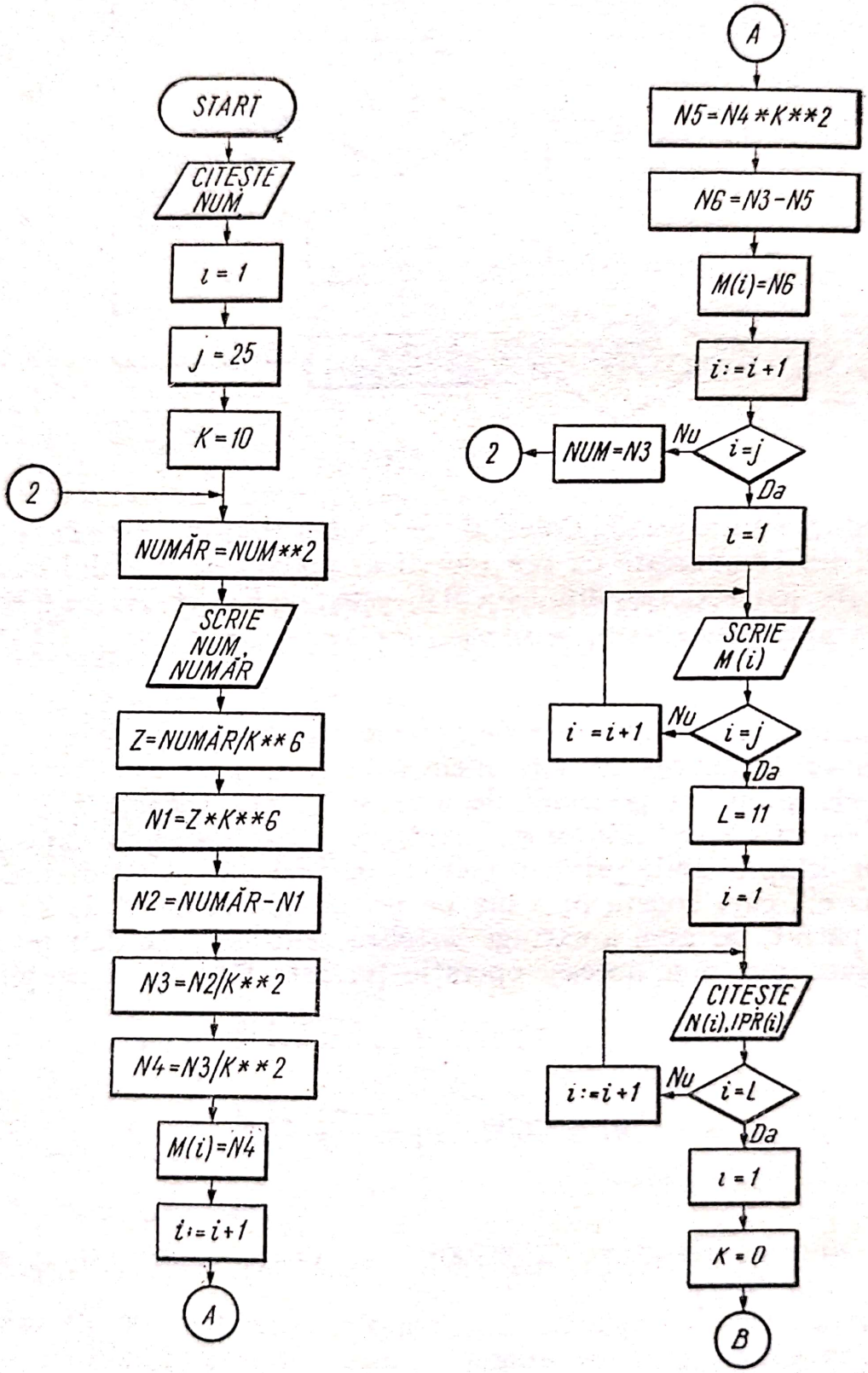
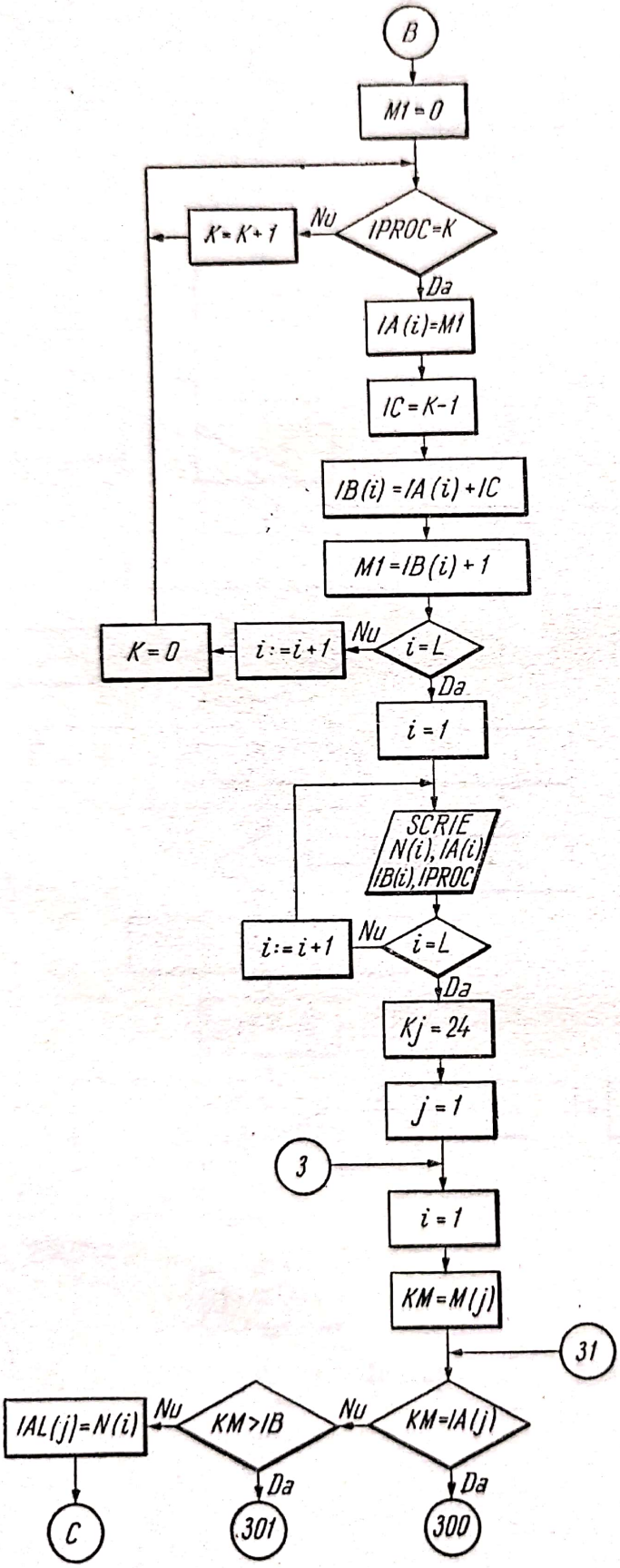
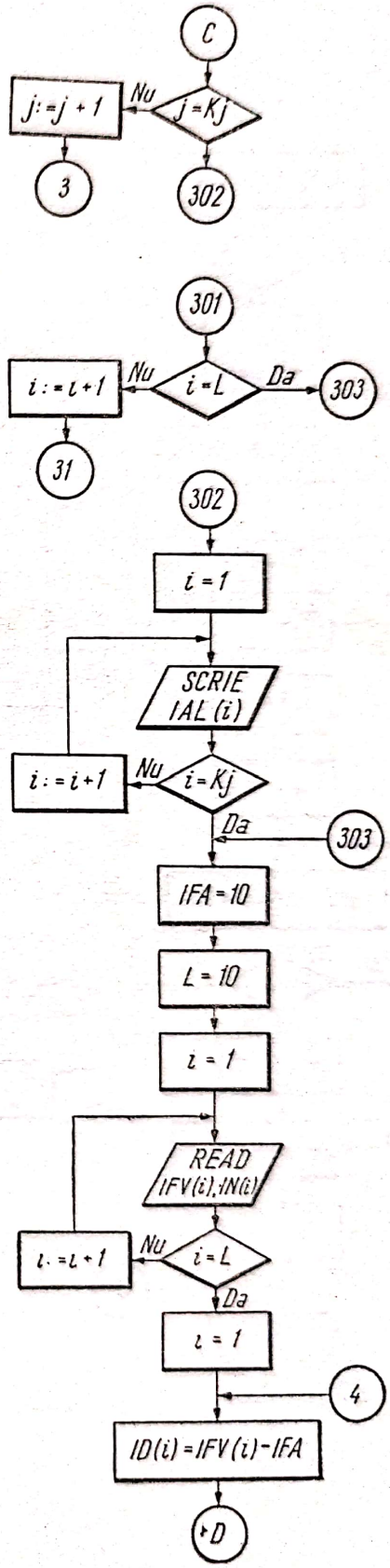


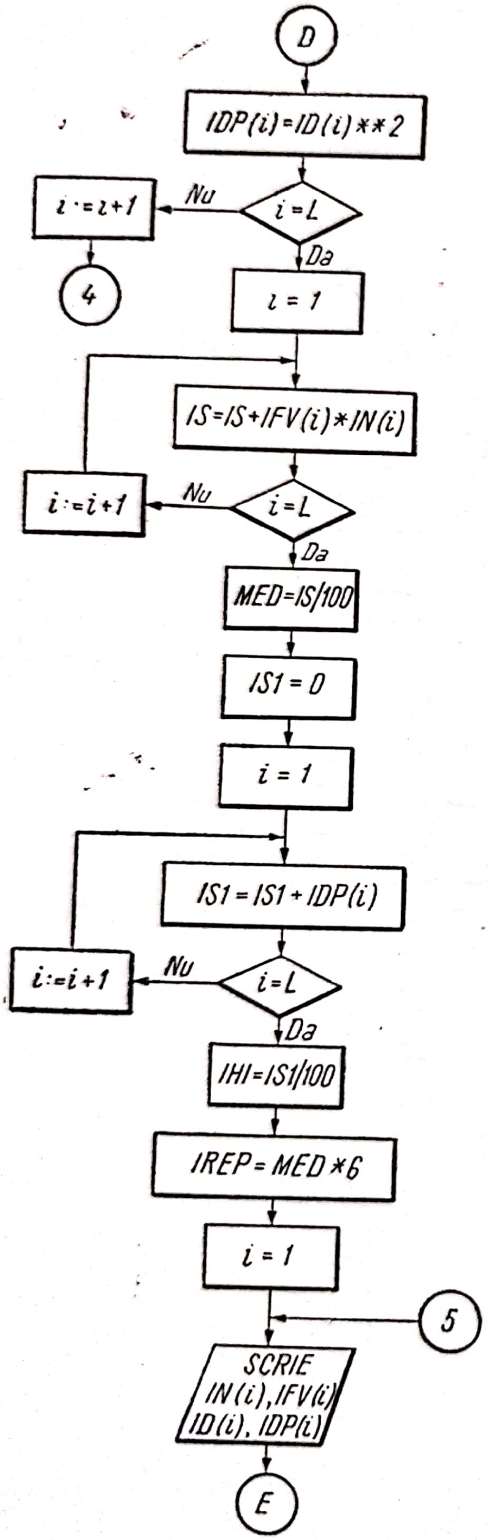
Fig. 9.21

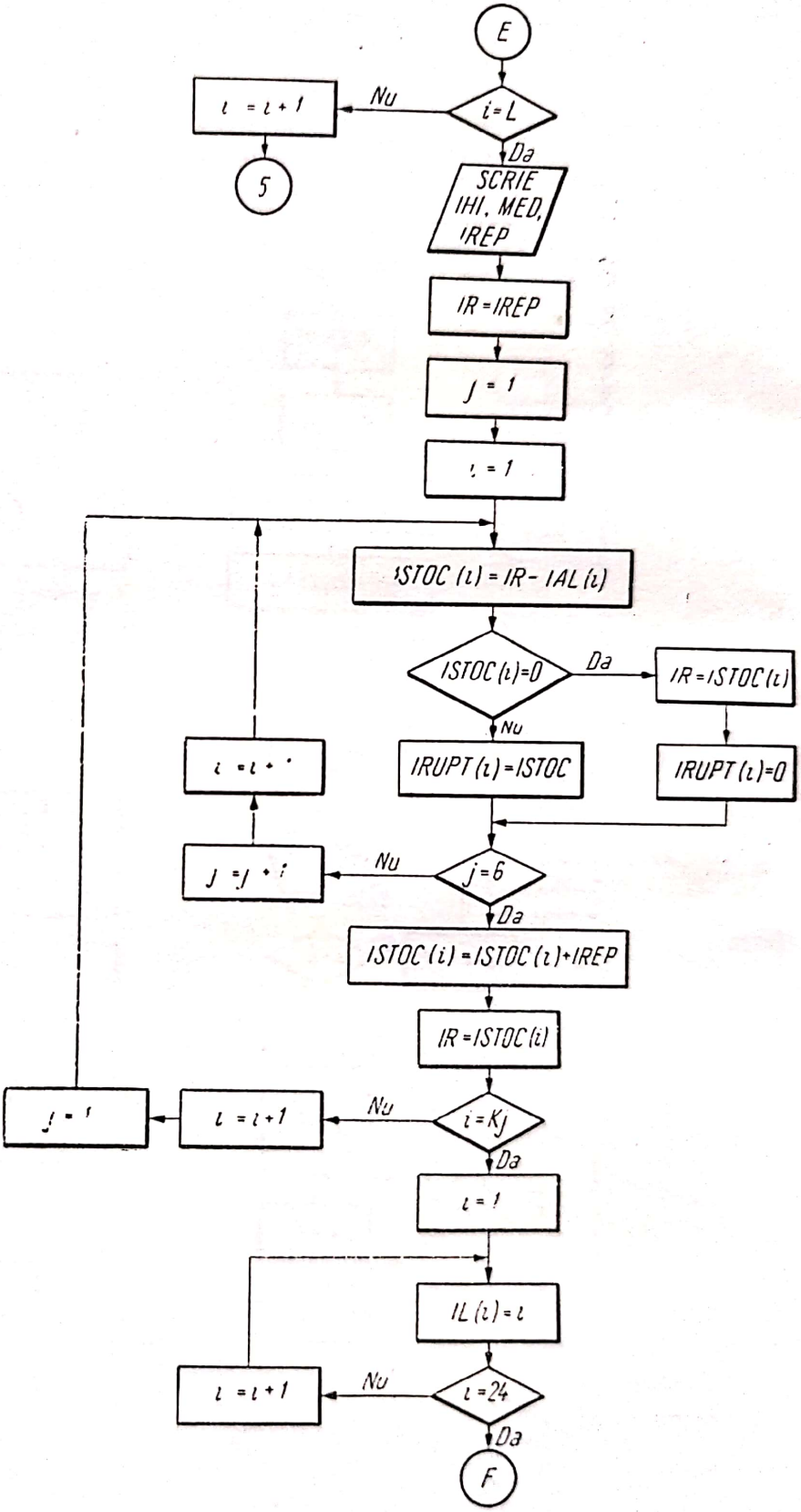
Scanned with OKEN Scanner



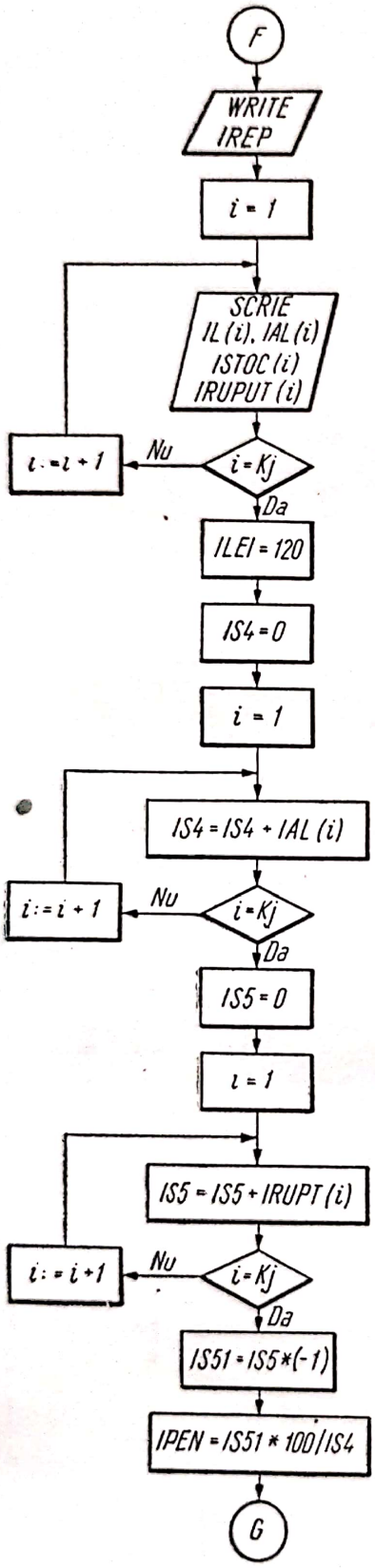


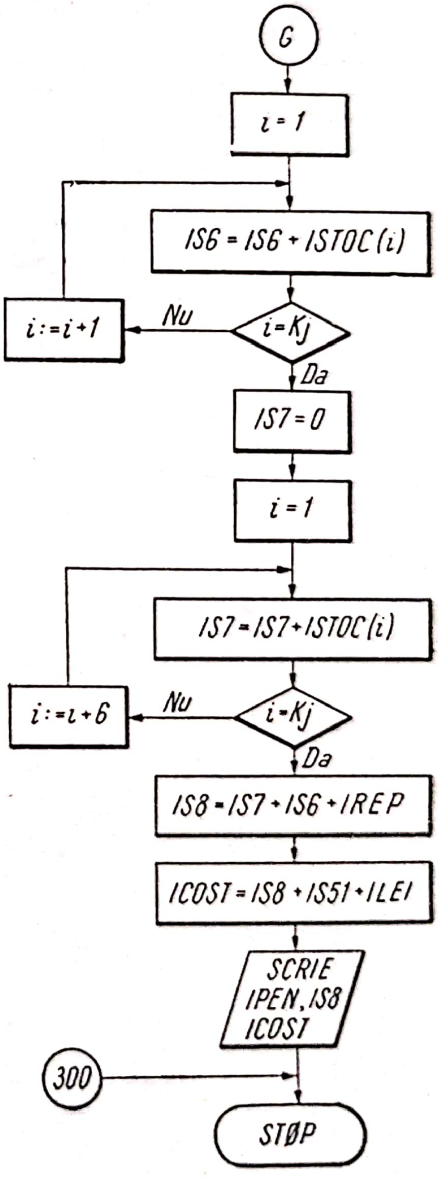
Scanned with OKEN Scanner





Scanned with OKEN Scanner





Scanned with OKEN Scanner

```
C     LISTA 9,5
C     ************************************
C     * PROGRAM FORTRAN PENTRU SIMULAREA *
C     * GESTIUNII STOCURILOR             *
C     ************************************
C
C       PROGRAM UTILIZAT IN CADRUL EX.9-2
C
C
C
C      PROGRAMUL DE SIMULARE A UNEI GESTIUNI
C      DE STOC CONTINE 4 PARTI:
C      -GENERAREA NUMERELOR ECHIPROBABILE
C      -GENERAREA ESANTIOANELOR
C      -CALCULUL MEDIEI FRECVENTEI
C      -SIMULAREA PROPRIUZISA
C
C      INTRODUCEREA DATELOR UTILIZATE IN
C      PROGRAM SE FACE DUPA URMATOARELE
C      MACHETE:
C      -MACHETA1 VARIABILA NUM I4 DIN COLOANA 1
C      -MACHETA2 VARIABILELE N(I),IPROC(I) 2I2
C       DIN COLOANA 1
C      -MACHETA3 VARIABILELE IFV(I),IN(I) I2,I1
C       DIN COLOANA 1
C
C      CU AJUTORUL PROGRAMULUI SE POATE SIMULA
C      PE O PERIOADA DE TIMP DORITA (IN CAZUL DE
C      FATA 24 ZILE LUCRATOARE .O LUNA) O GESTIUNE
C      DE STOC.
C       IN FUNCTIE DE RUPTURA DE STOC,DE ZILELE DE
C       STOCAJ SI DE COSTUL TOTAL IN FINAL SINT
C       ACCEPTATE ESANTIOANELE CERERII,IN CAZUL DE
C       FATA REAPROVIZIONAREA ESTE CONSTANTA SI SE
C       REALIZEAZA LA SFIRSITUL FIECAREI SAPTAMINI,
C
C
C
C
C
C
C
C
C
C
C      SEMNIFICATIA UNOR NUME DE VARIABILE UTILIZATE
C      IN ACEST PROGRAM ESTE URMATOAREA(SEMNUL= AVIND
C      SEMNIFICATIA DE ''REPREZINTA''):
C      IFV=FRECVENTA OBSERVATA
C      IHI=TESTUL HI PATRAT
C      MED=MEDIA DISTRIBUTIEI
C      IREP=CONSUMUL MEDIU
C      IL=ZIUA
C      IAL=CEREREA
C      ISTOC=STOCUL RAMAS DUPA SATISFACEREA CERERII
C      IRUPT=VALOAREA RUPTURI DE STOC
C      ILEI=COSTUL RUPTURII DE STOC
C      IPEN=RUPTURA DE STOC
C      IS8=ZILE STOCAJ
C      ICOST=VALOAREA TOTALA A GESTIUNII STOCULUI
C   ***********************************************************************
C   *    G E N E R A R E A   N U M E R E L O R   E C H I P R O B A B I L E   *
C   ***********************************************************************
```

```
      DIMENSION IFV(10),IN(10),ID(10),IDP(10),ISTOC(24),IKUPT(24),IL(24)
      DIMENSION N(100),IA(100),IB(100),IPROC(100),M(100),IA1(100)
      INTEGER Z
      WRITE(108,60)
   60 FORMAT('1',1X,47(1H*)//' ',1X,'* S I M U L A R E A  U N E I  G E S
     *T I U N I *'/' ',1X,1H*,45X,1H*/' ',1X,1H*,15X,'D E  S T O C',18X,
     *1H*/' ',1X,47(1H*)///)
      READ(105,1)NUM
    1 FORMAT(I4)
      I=1
      J=25
      K=10
      WRITE(108,61)
   61 FORMAT(' ',1X,18(1H*)//' ',1X,'*          *          *'/' ',1X,'* N(I) *
     * N(I)**2 *'/' ',1X,'*          *          *'/' ',1X,18(1H*)///)
    2 NUMAR=NUM**2
      WRITE(108,5)NUM,NUMAR
    5 FORMAT(' ',3X,I4,2X,I8,/)
      Z=NUMAR/K**6
      N1=Z*K**6
      N2=NUMAR-N1
      N3=N2/K**2
      N4=N3/K**2
      M(I)=N4
      I=I+1
      N5=N4*K**2
      N6=N3-N5
      M(I)=N6
      I=I+1
      IF(I.EQ.J)GO TO 100
      NUM=N3
      GO TO 2
  100 WRITE(108,62)
   62 FORMAT('1',1X,25(1H*)//' ',1X,1H*,21X,1H*/' ',1X,'* GENERAREA NUMER
     *ELOR *'/' ',1X,1H*,21X,1H*/' ',1X,'*   ECHIPROBABILE     *'/' ',1X
     *,1H*,21X,1H*/' ',1X,23(1H*)/)
      DO 4 I=1,J
    4 WRITE(108,3)M(I)
    3 FORMAT(' ',2X,I2,/)
C   **********************************************************
C   *    G E N E R A R E A  E S A N T I O A N E L O R    *
C   **********************************************************
      L=11
      DO 11 I=1,L
   11 READ(105,12)N(I),IPROC(I)
   12 FORMAT(2I2)
      I=1
      K=0
      M1=0
      WRITE(108,63)
   63 FORMAT('1',5X,'GENERAREA ESANTIOANELOR'//)
   16 IF(IPROC(I).EQ.K)GO TO 200
      K=K+1
      GO TO 16
  200 IA(I)=M1
      IC=K-1
      IB(I)=IA(I)+IC
      M1=IB(I)+1
      IF(I.EQ.L)GO TO 201
      I=I+1
      K=0
      GO TO 16
```


Scanned with OKEN Scanner

```
  201 WRITE(108,17)
   17 FORMAT(' ',6X,31(1H*)/' ',6X,'*        *        *        *        *'/' '
     *,6X,'* N(I) * IA(I) * IB(I) * P(X) *'/' ',6X,'*        *        *
     *    *        *'/' ',6X,31(1H*))
      DO 20 I=1,L
   20 WRITE(108,18)N(I),IA(I),IB(I),IPROC(I)
   18 FORMAT(' ',6X,1H*,2X,I2,2X,1H*,3X,I2,2X,1H*,3X,I2,2X,1H*,2X,I2,2X,
     *1H*)
      WRITE(108,19)
   19 FORMAT(' ',6X,31(1H*,////)
C     ***********************************************************
C     *    C A L C U L U L   M E D I E I   F R E C V E N T E I    *
C     ***********************************************************
      KJ=24
      J=1
      WRITE(108,64)
   64 FORMAT(' ',1X,16(1H*)/' ',1X,1H*,14X,1H*/' ',1X,'* ESANTIOANELE *'
     */' ',1X,1H*,14X,1H*/' ',1X,'*   CERERII    *'/' ',1X,1H*,14X,1H*/'
     * ',1X,16(1H*)/)
   30 I=1
      KM=M(J)
   31 IF(KM.LT.IA(I)) GO TO 300
      IF(KM.GT.IB(I)) GO TO 301
      IAL(J)=N(I)
      IF(J.EQ.KJ)GO TO 302
      J=J+1
      GO TO 30
  301 IF(I.EQ.L) GO TO 303
      I=I+1
      GO TO 31
  302 DO 32 I=1,KJ
   32 WRITE(108,33)IAL(I)
   33 FORMAT(' ',2X,I2)
      GO TO 600
  303 WRITE(108,34)
   34 FORMAT('1',2X,'M>IA>IB')
  600 IFA=10
      L=10
      DO 40 I=1,L
   40 READ(105,41)IFV(I),IN(I)
   41 FORMAT(I2,I1)
      DO 42 I=1,L
      ID(I)=IFV(I)-IFA
   42 IDP(I)=ID(I)**2
      IS=0
      DO 43 I=1,L
   43 IS=IS+IFV(I)*IN(I)
      MED=IS/100
      IS1=0
      DO 44 I=1,L
   44 IS1=IS1+IDP(I)
      IHI=IS1/10
      IREP=MED*6
      WRITE(108,65)
   65 FORMAT('1',5X,'CALCULUL  MEDIEI  DISTRIBUTIEI'//)
      WRITE(108,45)
   45 FORMAT(' ',6X,43(1H*)/' ',6X,'*        *              *              *
     * *    *'/' ',6X,'*        * FRECVENTA * FRECVENTA *    *  2 *'/' ',
     *6X,'* CIFRE *              *            *  D * D  *'/' ',6X,'*        *
     * OBSERVATA * TEORETICA *    *    *'/' ',6X,'*        *
     *          *    *    *'/' ',6X,43(1H*))
      DO 46 I=1,L
   46 WRITE(108,47)IN(I),IFV(I),ID(I),IDP(I)
   47 FORMAT(' ',6X,1H*,3X,I1,3X,1H*,4X,I2,5X,1H*,'    10      ',1H*,1X,I
```

```
      *2,1X,1H*,1X,I2,1X,1H*)
       WRITE(108,48)
   48 FORMAT(' ',6X,43(1H*)///)
       WRITE(108,49)IHI,MED,IREP
   49 FORMAT(' ',20X,'2'/' ',19X,'X=',I2/' ',19X,'MEDIA FRECVENTEI=',I2/
     *' ',19X,'CONSUMUL MEDIU=',I2)
       WRITE(108,66)
   66 FORMAT('1',5X,'SIMULAREA PROPRIUZISA A UNEI'/' ',9X,'GESTIUNI DE S
     *TOC'//)
C  ************************************************
C  *    S I M U L A R E A   P R O P R I U Z I S A    *
C  ************************************************
       IR=IREP
       J=1
       I=1
   50 ISTOC(I)=IR-IAL(I)
       IF(ISTOC(I).GE.0)GO TO 500
       IRUPT(I)=ISTOC(I)
   51 IF(J.EQ.6)GO TO 501
       J=J+1
       I=I+1
       GO TO 50
  500 IR=ISTOC(I)
       IRUPT(I)=0
       GO TO 51
  501 ISTOC(I)=ISTOC(I)+IREP
       IR=ISTOC(I)
       IF(I.EQ.KJ)GO TO 502
       I=I+1
       J=1
       GO TO 50
  502 DO 52 I=1,24
   52 IL(I)=I
       WRITE(108,53)IREP
   53 FORMAT(' ',6X,35(1H*)/' ',6X,'*          *          *          *          *'
     */' ',6X,'* ZIUA * CERFREA * STOC * RUPTURA *'/' ',6X,'*          *
     *          *          *'/' ',6X,35(1H*)/' ',6X,'*          *          *
     *',2X,I2,2X,1H*,'          *')
       DO 55 I=1,KJ
   55 WRITE(108,56)IL(I),IAL(I),ISTOC(I),IRUPT(I)
   56 FORMAT(' ',6X,1H*,2X,I2,2X,1H*,3X,I2,4X,1H*,2X,I2,2X,1H*,3X,I2,4X,
     *1H*)
       WRITE(108,57)
   57 FORMAT(' ',6X,35(1H*))
       ILEI=120
       IS4=0
       DO 70 I=1,KJ
   70 IS4=IS4+IAL(I)
       IS5=0
       DO 71 I=1,KJ
   71 IS5=IS5+IRUPT(I)
       IS51=IS5*(-1)
       IPEN=IS51*100/IS4
       IS6=0
       DO 72 I=1,KJ
   72 IS6=IS6+ISTOC(I)
       IS7=0
       DO 73 I=1,KJ,6
   73 IS7=IS7+ISTOC(I)
       IS8=IS7+IS6+IREP
       ICOST=IS8+IS51*ILEI
       WRITE(108,74)IPEN,IS8,ICOST
   74 FORMAT(' '//' ',8X,'RUPTURA DE STOC=',I3,' ',8X,'COSTUL STOCAJ
     *',I3/' ',8X,'COSTUL TOTAL PE'/' ',8X,'      26 DE ZILE=',I5/)
  500 STOP
       END
```

Lista 9.5

Scanned with OKEN Scanner

```
**************************************************
* S I M U L A R E A  U N E I  G E S T I U N I *
*                                                *
*                D E  S T O C                    *
**************************************************
```

```
********************
*      *           *
* N(I) * N(I)**2   *
*      *           *
********************

 8325   69272329
 2723    7414729
 4147   17197609
 1976    3904576
 9045   81812025
 8120   65934400
 9344   87310336
 3105    9628609
 6286   39513796
 5137   26388769
 3887   15108769
 1087    1181569
```

Lista 9.6.

```
************************
*                      *
* GENERAREA NUMERELOR  *
*                      *
*   ECHIPROBABILE      *
*                      *
************************
```

```
 27
 23
 41
 47
 19
 76
 90
 45
 81
 20
 93
 44
 51
  3
 62
 86
 51
 37
 38
 87
 10
 87
 18
 15
 **
```

Lista 9.7

Scanned with OKEN Scanner

GENERAREA ESANTIOANELOR

N(I)	IA(I)	IB(I)	P(%)
0	0	2	3
1	3	7	5
2	8	17	10
3	18	37	20
4	38	67	30
5	68	77	10
6	78	85	8
7	86	90	5
8	91	94	4
9	95	97	3
10	98	99	2

Lista 9.8

ESANTIOANELE
CERERII

3
3
4
4
3
5
7
4
6
3
8
4
3
1
4
7
4
3
4
7
2
7
3
2

CALCULUL MEDIEI DISTRIBUTIEI

CIFRE	FRECVENTA OBSERVATA	FRECVENTA TEORETICA	D	D	2
0	16	10	6	36	
1	10	10	0	0	
2	10	10	0	0	
3	9	10	-1	1	
4	8	10	-2	4	
5	9	10	-1	1	
6	12	10	2	4	
7	11	10	1	1	
8	12	10	2	4	
9	3	10	-7	49	

2
X=10
MEDIA FRECVENTEI= 4
CONSUMUL MEDIU=24

Lista 9.9

Scanned with OKEN Scanner

SIMULAREA PROPRIUZISA A UNEI
GESTIUNI DE STOC

ZIUA	CEREREA	STOC	RUPTURA
		24	
1	5	19	0
2	2	17	0
3	2	15	0
4	3	12	0
5	3	9	0
6	4	29	0
7	4	25	0
8	3	22	0
9	3	19	0
10	7	12	0
11	3	9	0
12	1	32	0
13	3	29	0
14	3	26	0
15	4	22	0
16	2	20	0
17	4	16	0
18.	2	38	0
19	3	35	0
20	6	29	0
21	3	26	0
22	4	22	0
23	5	17	0
24	4	37	0

RUPTURA DE STOC= 0
ZILE STOCAJ =669
COSTUL TOTAL PTR
26 DE ZILE= 669

Lista 9.10

SIMULAREA PROPPIUZISA A UNEI
GESTIUNI DE STOC

ZIUA	CEREREA	STOC	RUPTURA
		27	
1	3	21	0
2	3	18	0
3	4	14	0
4	4	10	0
5	3	7	0
6	5	26	0
7	7	19	0
8	4	15	0
9	6	9	0
10	3	6	0
11	8	-2	-2
12	4	26	0
13	3	23	0
14	1	22	0
15	4	18	0
16	7	11	0
17	4	7	0
18	3	28	0
19	4	24	0
20	7	17	0
21	2	15	0
22	7	8	0
23	3	5	0
24	2	27	0

RUPTURA DE STOC= 1
ZILE STOCAJ =485
COSTUL TOTAL PTR
26 DE ZILE= 725

Lista 9.11

Scanned with OKEN Scanner

Avînd şirul de numere echiprobabile şi grupîndu-le cîte două, cu ajutorul tabelului „Generarea eşantioanelor" (v. lista 9.8), se generează eşantioanele cererii; de exemplu: 70 se află între 68 şi 77, deci eşantionul cererii este 5.

Fragmentul de tabelă de mai jos, adică:

$N(I)$	$IA(I)$	$IB(I)$	$P(\%)$
.	.	.	.
.	.	.	.
.	.	.	.
5	68	77	10
.	.	.	.
.	.	.	.
.	.	.	.

este construit pe baza probabilităţii cererii.

În exemplul de faţă, dacă $P(\%) = 10$ iar limita superioară este 68, limita inferioară va fi peste zece, adică 77 ş.a.m.d. În acest mod se generează şirul de numere reprezentînd eşantioanele cererii.

2° *Elementele iniţiale ale simulării*. Pentru a începe simularea, avem nevoie de anumite elemente iniţiale pe care le fixăm arbitrar, dar în aşa fel încît să se producă cît mai puţine oscilaţii tranzitorii ce pot duce la estimări de date false.

Stocul iniţial de plecare este cel calculat anterior, şi anume consumul mediu. În funcţie de cerere şi de nivelul stocului se calculează nivelul rămas şi, dacă este cazul, ruptura de stoc.

Cererea de ruptură de stoc este dată de raportul:

$$\left(\sum_{1}^{n} \textit{ruptura de stoc} / \sum_{1}^{n} \textit{cerere}\right) * 100 \ (\%).$$

Costul total asupra celor n zile alese pentru simulare se calculează adunînd la numărul zilelor de stocaj, cantitatea rupturii de stoc înmulţită cu costul K_p.

Dacă costul global nu satisface dorinţele, se pot repeta calculele, cu ajutorul aceluiaşi program, pînă la obţinerea unei situaţii acceptabile.

9.3. APLICAŢIILE TEORIEI ECHIPAMENTULUI

Avînd în vedere că fiecare unitate productivă este dotată cu o varietate largă de utilaje ce lucrează în diferite condiţii de lucru, precum şi cu moduri diferite de comportare la uzură, s-au efectuat intense studii pentru găsirea unor metode adecvate privind întreţinerea, repararea şi înlocuirea acestor utilaje, studii care au dus la constituirea unei noi teorii în cercetarea operaţională: *teoria echipamentelor*.

Ansamblul măsurilor ce sînt luate pentru bune funcţionare a utilajelor din dotare, defineşte politica de utilizare, întreţinere şi înlocuire a utilajelor unei întreprinderi, în procesul de producţie.

Scanned with OKEN Scanner

La stabilirea unei astfel de politici se au în vedere problemele privind economicitatea.

De aceea, dintre toate soluţiile ce se găsesc privind o bună politică de întreţinere sau înlocuire, trebuie aleasă, prin diferite metode, cea optimă, adică soluţia care reprezintă în final un „compromis" între siguranţa în funcţionare şi cheltuielile ce se cer a fi minime.

În cadrul unei politici astfel alease, principalele acţiuni care o compun pot fi următoarele:

— întreţinerea curentă,
— înlocuirea sistematică a pieselor ce se uzează mai repede,
— repararea pieselor uzate,
— verificări periodice, chiar dacă utilajul este în stare bună de funcţionare, pentru prevenirea unor defecţiuni ce ar dăuna procesului de producţie.

Înainte de a face cîteva consideraţii matematice cu privire la aplicaţiile cercetării operaţionale în acest domeniu, trebuie precizaţi cîţiva termeni cu care lucrează teoria echipamentelor:

— *viaţa unui echipament* reprezintă intervalul cuprins între punerea în funcţiune a echipamentului şi scoaterea lui din funcţiune ca urmare a uzurii totale;

— *uzura totală* reprezintă momentul în care echipamentul este scos definitiv din funcţiune;

— *vîrsta unui echipament* reprezintă timpul scurs din momentul punerii în funcţiune pînă în momentul cînd se face observaţia;

— *curba de supravieţuire* care constă în reprezentarea grafică a probabilităţii de supravieţuire în timp.

Probabilităţi şi curbe de supravieţuire

Pentru început, să examinăm, plecînd de la o mulţime oarecare de piese de schimb de acelaşi tip, curba de supravieţuire, a cărei alură determină rata de reaprovizionare cu piese de schimb.

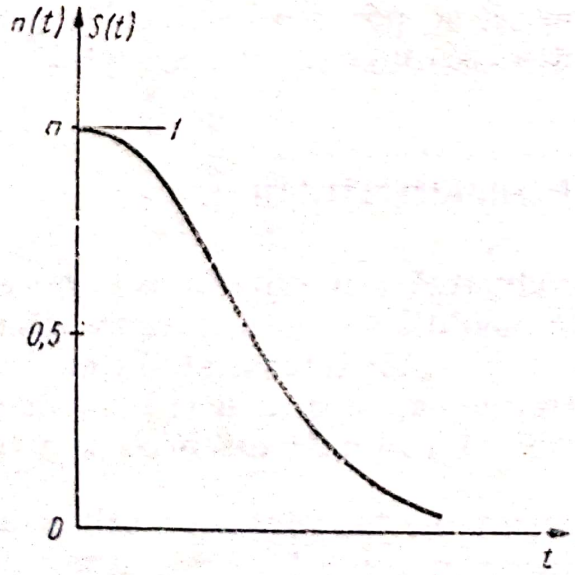


Fig. 9.22

Datorită funcţionării în timp, mulţimea pieselor va scade pînă la zero în funcţie de caracteristicile pieselor la uzură.

Notînd cu t clasa de timp şi cu $n(t)$ numărul pieselor rămase în funcţiune la fiecare dată t obţinem foarte uşor curba n funcţie de timp (fig. 9.22).

Făcînd, în continuare, raportul: $\frac{n(t)}{n(0)}$, unde $n(0)$ reprezintă numărul total de piese intrate în funcţiune, obţinem *probabilitatea de supravieţuire* a unei piese pe toată perioada determinată:

$$S(t) = \frac{n(t)}{n(0)}.$$

Scanned with OKEN Scanner

De exemplu, dacă la $n(0) = 1000$ bucăţi pinioane intrate în funcţiune, probabilitatea ca un pinion să dureze mai puţin de 6 luni este:

$$\frac{n(6)}{n(0)} = \frac{850}{1000} = 0{,}85,$$

unde $n(6) = 850$ reprezintă numărul de piese rămase în luna a 6-a în funcţiune, număr determinat statistic.

Probabilitatea ca un pinion să înceteze de a mai fi în funcţiune într-o perioadă de timp de exemplu $(t-1,\ t)$ este dată de relaţia:

$$P_t = \frac{n(t-1) - n(t)}{n(0)} = S(t-1) - S(t):$$

Astfel, probabilitatea ca un pinion să înceteze de a mai funcţiona între luna 10 şi 11 este:

$$P_{11} = \frac{n(10) - n(11)}{n(0)}.$$

Dacă considerăm intervalul de timp $(t, t + \Delta t)$ atunci:

$$P_t = S(t) - S(t + \Delta t),$$

$$\frac{P_t}{\Delta t} = \frac{S(t) - S(t - \Delta t)}{\Delta t} = S'(t),$$

care reprezintă prin diferenţe finite derivate funcţiei $S(t)$; atunci:

$$P_t = - S'(t) \cdot \Delta t.$$

Este interesant să cunoaştem *probabilitatea de avarie*, $p_c(t)$, altfel spus probabilitatea ca un utilaj aflat în stare de funcţionare pînă la timpul $t-1$, să înceteze a mai funcţiona din motive de „pană", în intervalul de timp $(t-1,\ t)$.

Pentru a calcula această probabilitate de avarie, remarcăm că probabilitatea P_t ca un utilaj să înceteze a mai funcţiona în intervalul de timp $(t-1,\ t)$ reprezintă produsul dintre probabilitatea de supravieţuire pînă la timpul $t-1$, adică $S(t-1)$, şi probabilitatea de avarie, P_c, la timpul t:

$$P_t = S(t-1) \cdot p_c(t),$$

de unde rezultă:

$$p_c(t) = \frac{P_t}{S(t-1)} = \frac{n(t-1) - n(t)}{n(0)} \cdot \frac{n(0)}{n(t-1)},$$

$$p_c(t) = 1 - \frac{n(t)}{n(t-1)}.$$

Stabilirea acestor probabilităţi presupune observaţii şi măsurători statistice concrete, ele servind la calcularea unor parametri cum ar fi:

— vîrsta medie de apariţie a unei avarii:

$$\bar{t} = \sum_1^{\infty} t.\ p_t;$$

Scanned with OKEN Scanner

— abaterea medie pătrată:

$$\sigma^2 = \sum_1^\infty t^2 \cdot p_t - \bar{t}^2 \quad \text{sau} \quad \sigma = \sqrt{\sum_1^\infty t^2 \cdot p_t - \bar{t}^2}.$$

Pentru $\bar{t}$ se mai poate da și o altă exprimare și anume:

$$t \cdot p_t = t[S(t-1) - S(t)] = t \cdot \Delta S.$$

Privind figura 9.23, dacă facem $\Sigma t \cdot p_t$ pe toată perioada de timp t, suprafața este mult mai mare decît suprafața descrisă de $S(t)$. În cazul cînd funcția $S(t)$ este continuă și ΔS tinde către zero obținem suprafața descrisă de curba de supraviețuire. Putem deci să scriem:

$$\bar{t} = \int_0^\infty S(t)\ \mathrm{d}t.$$

Avînd în vedere considerentele făcute pînă aici, vom studia în continuare cele două cazuri limită.

Probabilitatea de ruptură constantă se referă la cazul în care piesele ies din funcțiune la intervale de timp constante datorită unei oboseli totale. Asupra acestor piese nu apare nici un interes de a face o întreținere preventivă. Curba de supraviețuire în acest caz are o alură exponențială (fig. 9.24).

În acest caz funcția de supraviețuire are forma:

$$S(t) = \mathrm{e}^{-\alpha t}$$

și vîrsta medie este:

$$\bar{t} = \int_0^\infty \mathrm{e}^{-\alpha t}\ \mathrm{d}t = \frac{1}{\alpha}.$$

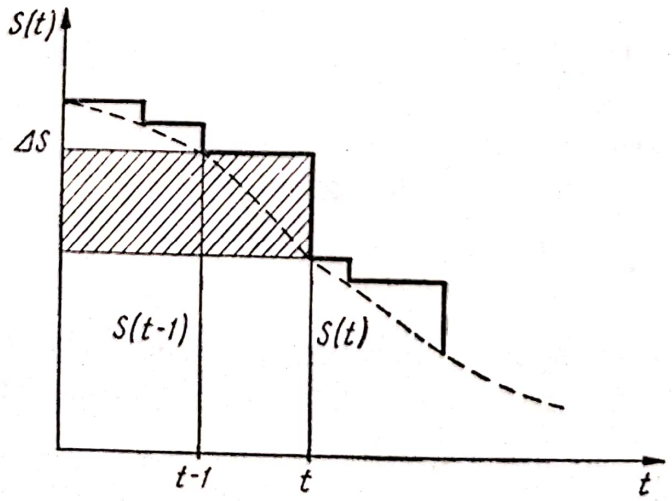


Fig. 9.23

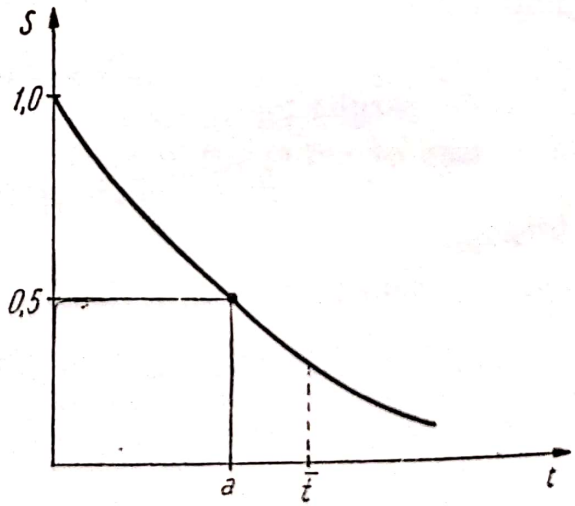


Fig. 9.24

Scanned with OKEN Scanner

Dacă considerăm valoarea de 0,5 a lui $S(t)$ (v. fig. 9.24):

$$S(t) = e^{-\alpha t} = 0{,}5,$$

şi notăm pentru acest caz $t = a$:

$$\frac{1}{2} = e^{-\alpha a},$$

$$\ln \frac{1}{2} = -\alpha a,$$

$$-\ln 2 = -\alpha a,$$

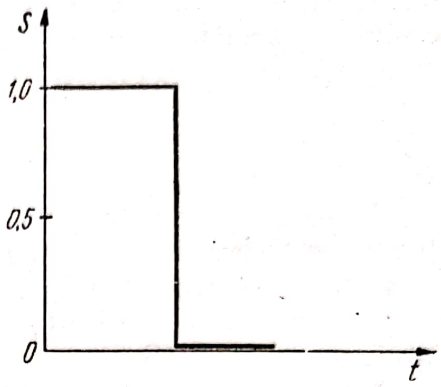


Fig. 9.25

atunci:

$$a = \frac{\ln 2}{\alpha} = \ln 2 \cdot \frac{1}{\alpha}.$$

De aici se vede că pentru acest gen de piese este caracteristic ca $a < \bar{t}$.

Piesele de uzură, care sînt piesele ce constituie un lot omogen, au o comportare identică la uzură.

Curba de supravieţuire în acest caz are o formă aproape dreptunghiulară, aşa ca în figura 9.25. Cu cît curba de supravieţuire se apropie de o formă dreptunghiulară, cu atît probabilitatea de avarie care este nulă pînă la $a = \bar{t}$, va avea un salt brusc către valoarea 1.

Cazul acestor piese de uzură nu este interesant pentru studiu deoarece toate elementele necesare întreţinerii se determină uşor şi precis.

Interesant este de ştiut, în cazul unui număr de echipamente în funcţiune, care este necesarul de reaprovizionat, pe o perioadă de timp, în funcţie de nevoile de înlocuire a pieselor uzate.

Aceasta reprezintă tocmai *rata de reaprovizionare* pe o perioadă de timp, cu ajutorul căreia se poate trasa o curbă de reaprovizionare.

Un exemplu

Ne propunem, spre exemplificare, să determinăm cantitatea optimă de piese de schimb necesară reaprovizionării pe o perioadă de timp stabilită în aşa fel încît numărul echipamentelor aflate în funcţiune să fie mereu acelaşi.

Am presupus că avem în funcţiune la timpul zero 100 de pompe, iar pe parcurs pompele „cad în pană" conform tabelei:

Timpul	0	1	2	3	4	5
Numărul de pompe în funcţiune	100	90	60	20	2	0

Scanned with OKEN Scanner

întocmită pe baza unor observații statistice. Cu valorile din această tabelă calculăm, pentru fiecare interval de timp, probabilitatea de avarie $p_c(t)$ cu relația:

$$p_c(t) = 1 - \frac{n(t)}{n(t-1)}.$$

Pentru rezolvarea problemei utilizăm un sistem de calcul automat. Mersul calculului este redat în schema logică din figura 9.26, pe baza căreia s-a întocmit și programul FORTRAN de prelucrare automată (lista 9.12). Rezultatele obținute, pe care le vom comenta în continuare, sînt redate în lista 9.13.

Mai întîi se calculează, cu relația dată mai sus, probabilitatea de avarie $p_c(t)$, imprimată în primul tabel al listei cu rezultate sub numele $PC(T)$ din coloana a treia (v. lista 9.13).

În continuare întocmim graful corespunzător valorilor lui $p_c(t)$ pe fiecare interval de timp (fig. 9.27).

Cu ajutorul acestui graf se determină o matrice **M** care, în cazul exemplului considerat, este:

$$\mathbf{M} = \begin{bmatrix} 1/10 & 9/10 & 0 & 0 & 0 \\ 1/3 & 0 & 2/3 & 0 & 0 \\ 2/3 & 0 & 0 & 1/3 & 0 \\ 9/10 & 0 & 0 & 0 & 1/10 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix};$$

și această matrice se calculează tot prin program (v. lista 9.13, matricea de sub titlul „TRANSPUNEREA GRAFULUI ÎNTR-O MATRICE").

Se calculează, în continuare, un vector de stare:

$$\mathbf{P}_{n+1} = \mathbf{P}_n \cdot \mathbf{M}.$$

Pentru o mai bună aproximare se alege n mai mare decît dimensiunea metrică (în exemplul considerat $n = 9$) și rezultă:

$$\begin{aligned} \mathbf{P}_1 &= \mathbf{P}_0 \cdot \mathbf{M} \\ \mathbf{P}_2 &= \mathbf{P}_1 \cdot \mathbf{M} \\ &\cdot \quad \cdot \quad \cdot \\ &\cdot \quad \cdot \quad \cdot \\ &\cdot \quad \cdot \quad \cdot \\ \mathbf{P}_9 &= \mathbf{P}_8 \cdot \mathbf{M} \end{aligned}$$

(suma termenilor din vectorul de stare $\mathbf{P}_{n+1}$ trebuie să fie egală cu numărul total de echipamente la timpul zero, în cazul exemplului 100). Vectorul de stare, sub numele PN este, de asemenea, redat în lista cu rezultate (v. lista 9.13).

Cu primele valori din fiecare vector de stare se trasează grafic curba de reaprovizionare, care are forma din figura 9.28 (în acest grafic sînt transpuse, de fapt, valorile obținute prin calcule analitice — efectuate cu programul din lista 9.12 și redate numeric în lista 9.13 cu rezultate, sub

Scanned with OKEN Scanner

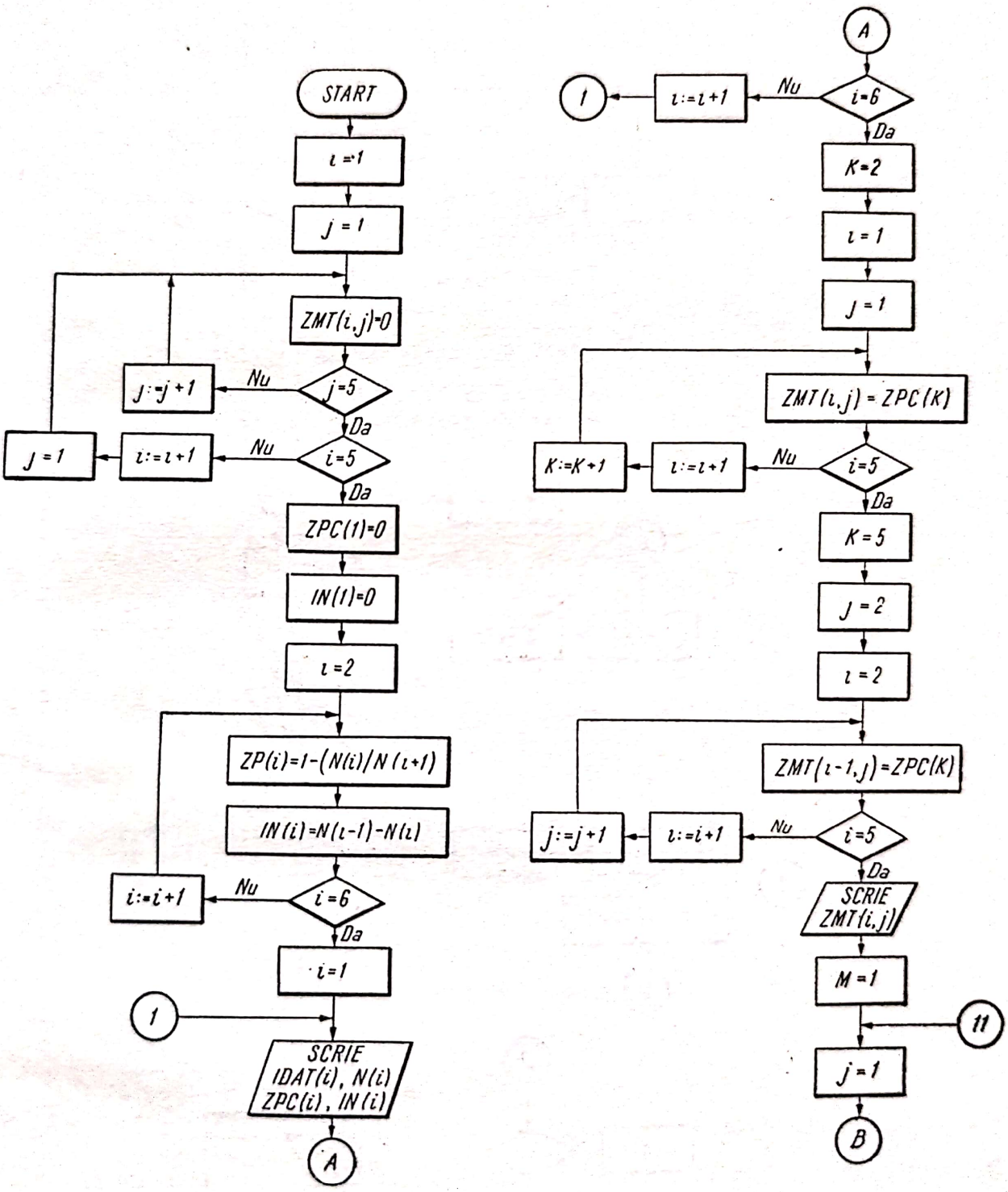
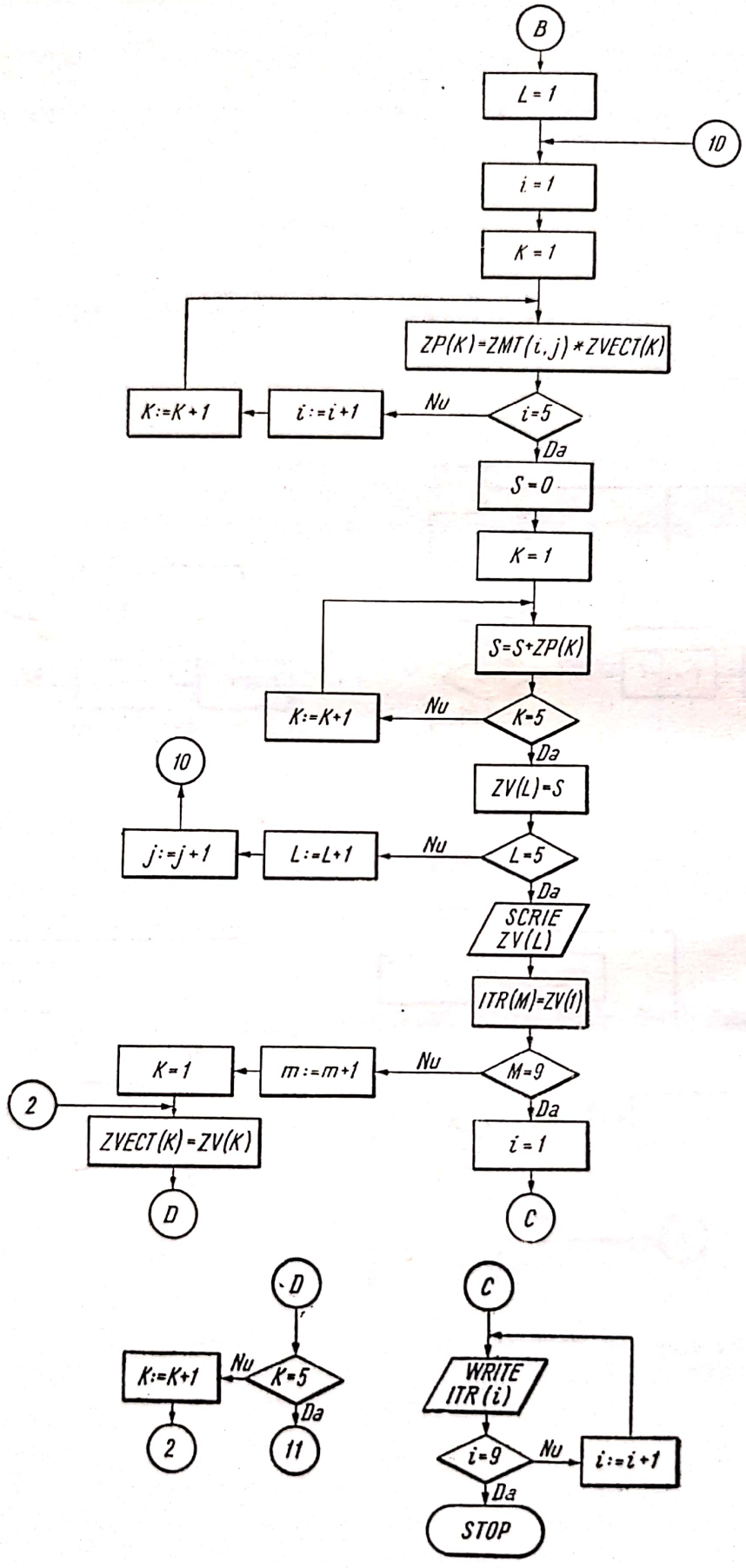


Fig. 9.26

Scanned with OKEN Scanner



Scanned with OKEN Scanner

```
C      LISTA 9.12
C      ****************************************
C      * PROGRAM FORTRAN PENTRU CALCULE DE    *
C      * INLOCUIRE A ECHIPAMENTELOR UZATE     *
C      ****************************************
C
C
C
C
C
C
C
C
C
C
C
C
C
C       PROGRAMUL DETERMINA IN FINAL VECTORUL
C       CURBEI DE REAPROVIZIONARE,PE BAZA OB-
C       SERVATIEI PRIVIND CADEREA ECHIPAMENTELOR
C       IN PANA.
C        DATELE INTRODUSE IN PROGRAM :
C        -NUMARUL ECHIPAMENTELOR LUATE IN
C        OBSERVATIE.
C        -PROBABILITATEA DE AVARIE IN TIMP.
C
C       DATELE AU FOST INTRODUSE IN PROGRAM
C       CU DATA DEOARECE NUMARUL ECHIPAMENTELOR
C       OBSERVATE ESTE REDUS.
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C      SEMNIFICATIA UNOR NUME DE VARIABILE UTILIZATE
C      IN ACEST PROGRAM ESTE URMATOAREA(SEMNUL= AVIND
C      SEMNIFICATIA DE ''REPREZINTA''):
C      NE=NUMARUL TOTAL DE ECHIPAMENTE
C      ZMT=MATRICEA GRAFULUI
C      ZPC=PROBABILITATEA DE AVARIE
C      IDAT=DATA
C      IN=NUMARUL DE ECHIPAMENTE CAZUTE IN PANA
C      ZVECT=VECTORUL DE STARE
C      INTR=VECTORUL DE TRASARE A GRAFICULUI
       REAL NE(6)
       DIMENSION ZMT(5,5),ZPC(6),IDAT(6),IN(6)
       DIMENSION ZP(5),ZVECT(5),ZV(6),ITR(9)
       DATA IDAT/0,1,2,3,4,5/,NE/100,90,60,20,2,0/,ZVECT/100.,4*0./
       WRITE(108,22)
    22 FORMAT('1',9X,'DETERMINAREA CANTITATII OPTIME DE'/' ',8X,'PIESE DE
      * SCHIMB PTR. REAPROVIZIONAT'//)
       DO 1 I=1,5
       DO 1 J=1,5
     1 ZMT(I,J)=0.
       ZPC(1)=0.
       IN(1)=0
       WRITE(108,20)
    20 FORMAT(' ',6X,43(1H*)/' ',6X,1H*,6X,1H*,12X,1H*,7X,1H*,13X,1H*/' '
      *,6X,'* DATA * NR.ECHIPA- * PC(T) * N(T-1)-N(T) *'/' ',6X,1H*,6X,'*
```

Scanned with OKEN Scanner

```
      *   MENTE    *',7X,1H*,13X,1H*/' ',6X,43(1H*))
       DO 2 I=2,6
       ZPC(I)=1,-(NE(I)/NE(I-1))
     2 IN(I)=NE(I-1)-NE(I)
       DO 3 I=1,6
     3 WRITE(108,4)IDAT(I),NE(I),ZPC(I),IN(I)
     4 FORMAT(' ',6X,1H*,3X,I1,2X,1H*,5X,I3,4X,1H*,2X,F4.2,1X,1H*,5X,I2,6
      *X,1H*)
       WRITE(108,21)
    21 FORMAT(' ',6X,43(1H*)//)
       K=2
       I=1
       J=1
     6 ZMT(I,J)=ZPC(K)
       IF(I.EQ.5)GO TO 100
       I=I+1
       K=K+1
       GO TO 6
   100 K=5
       J=2
       I=2
     7 ZMT(I-1,J)=ZPC(K)
       IF(I.EQ.5)GO TO 101
       I=I+1
       J=J+1
       K=K-1
       GO TO 7
   101 WRITE(108,23)
    23 FORMAT(' ',7X,'TRANSPUNEREA GRAFULUI'/' ',11X,'INTR-O MATRICE'//)
       WRITE(108,8)((ZMT(I,J),J=1,5),I=1,5)
     8 FORMAT(5(6X,F4.2,2X))
       WRITE(108,30)
    30 FORMAT(' ')
C PRODUSUL A UNEI MATRICI CU UN VECTOR
       M=1
       WRITE(108,24)
    24 FORMAT(' ',9X,'DETERMINAREA VECTORILOR DE STARE PN'//)
    15 J=1
       L=1
    11 I=1
       K=1
     9 ZP(K)=ZMT(I,J)*ZVECT(K)
       IF(I.EQ.5)GO TO 102
       I=I+1
       K=K+1
       GO TO 9
   102 S=0
       DO 10 K=1,5
    10 S=S+ZP(K)
       ZV(L)=S
       IF(L.EQ.5) GO TO 103
       L=L+1
       J=J+1
       GO TO 11
   103 WRITE(108,12)(ZV(L),L=1,5)
    12 FORMAT(' ',1X,/' ',6X,5(F5.2,2X))
       ITR(M)=ZV(1)
       IF(M.EQ.9)GO TO 104
       M=M+1
       DO 14 K=1,5
    14 ZVECT(K)=ZV(K)
       GO TO 15
   104 WRITE(108,25)
    25 FORMAT(' ',6X,'CURBA DE'/' ',3X,'REAPROVIZIONARE'//)
       DO 16 I=1,9
    16 WRITE(108,17)ITR(I)
    17 FORMAT(' ',6X,I2)
       STOP
       END
```

Lista 9.12

Scanned with OKEN Scanner

DETERMINAREA CANTITATII OPTIME DE
PIESE DE SCHIMB PTR. REAPROVIZIONAT

DATA	NR. ECHIPAMENTE	P(T)	N(T-1)-N(T)
0	100	.00	0
1	90	.10	10
2	60	.33	30
3	20	.67	40
4	2	.90	18
5	0	1.00	2

TRANSPUNEREA GRAFULUI
INTR-O MATRICE

.10	.90	.00	.00	.00
.33	.00	.67	.00	.00
.67	.00	.00	.33	.00
.90	.00	.00	.00	.10
1.00	.00	.00	.00	.00

DETERMINAREA VECTORILUR DE STARE PN

10.00	90.00	.00	.00	.00
31.00	9.00	60.00	.00	.00
46.10	27.90	0.00	20.00	.00
35.91	41.49	18.60	2.00	2.00
33.62	32.32	27.66	6.60	.20
38.36	30.26	21.55	9.62	.62
37.20	34.52	20.17	7.18	.92
36.06	33.48	23.01	6.72	.72
36.88	32.46	22.32	7.67	.67

CURBA DE
REAPROVIZIONARE

10
31
46
35
33
38
37
36
36

STOP

Lista 9.13

Scanned with OKEN Scanner

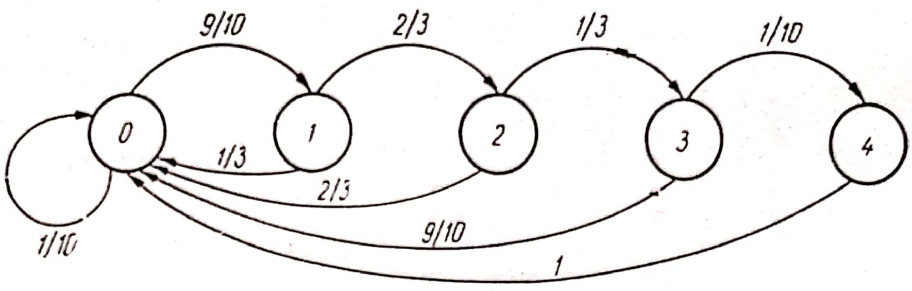


Fig. 9.27

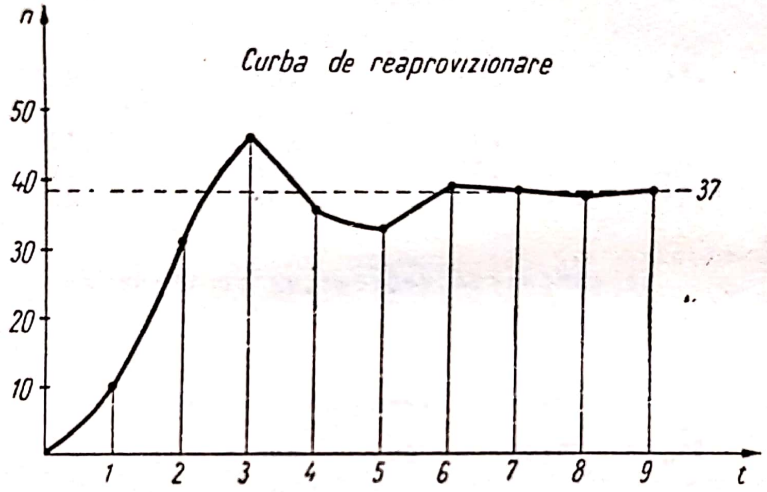


Fig. 9.28

numele „CURBA DE REAPROVIZIONARE"). Constatăm că numărul de piese de reaprovizionat este în jurul lui 37.

Pentru determinarea valorii limită a acestei curbe verificăm că graful este tare conex, adică:

$$\mathbf{P}^* = \mathbf{P}^* \cdot \mathbf{M},$$

unde $\mathbf{P}^*$ reprezintă vectorii de stare optimali. Rezultă în continuare:

$$[P_0^* \; P_1^* \; P_2^* \; P_3^* \; P_4] = [P_0^* \; P_1^* \; P_2^* \; P_3^* \; P_4] \cdot \begin{array}{c} \begin{matrix} a & b & c & d & e \end{matrix} \\ \begin{bmatrix} 1/10 & 9/10 & 0 & 0 & 0 \\ 1/3 & 0 & 2/3 & 0 & 0 \\ 2/3 & 0 & 0 & 1/3 & 0 \\ 9/10 & 0 & 0 & 0 & 1/10 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \begin{matrix} a \\ b \\ c, \\ d \\ e \end{matrix}$$

Scanned with OKEN Scanner

cu ajutorul căruia obținem în continuare, următorul sistem de ecuații:

$$\begin{cases} a+b+c+d+e=100 \\ \frac{a}{10}+\frac{b}{3}+\frac{2c}{3}+\frac{9d}{10}+e=a \\ \frac{9a}{10}=b \\ \frac{2b}{3}=c \\ c/3=d \\ d/10=e \end{cases}$$

Soluțiile acestui sistem sînt următoarele:

$$a=36{,}7\simeq 37$$
$$b=33{,}2\simeq 33$$
$$c=22{,}1\simeq 22$$
$$d=7{,}36\simeq 7$$
$$e=0{,}74\simeq 1$$

Observăm că valoarea cea mai mare este 37.

În concluzie, cantitatea de piese necesare reaprovizionării pentru a avea în funcțiune mereu cele 100 de pompe este în jurul lui 37.

Scanned with OKEN Scanner

BIBLIOGRAFIE

1. Bellman, R., Dreyfus, S. *Programarea dinamică aplicată.* Bucureşti, Editura tehnică, 1967.
2. Boiarinov, I. A., Kafarov, V. V. *Metodî optimizaţii v himiceskoi tehnologhii.* Moskva, Izdatelstvo Himia, 1969.
3. Dantzig, G. *Linear Programming and Extensions.* New York, Princeton University Press, 1963.
4. Desbazeille, G. *Exercices et problèmes de recherche opérationnelle.* Paris, Dunod, 1972.
5. Dragomirescu, M., Maliţa, M. *Programarea pătratică.* Bucureşti, Editura ştiinţifică, 1968.
6. Drăghici, Gh. *Bazele proiectării proceselor tehnologice în industria construcţiilor de maşini.* Bucureşti, Editura tehnică, 1973.
7. Dumitrescu, St., Dumitrescu, I., Paiu, O. *Aplicaţii inginereşti ale calculatoarelor, cu exemple din industria petrolieră şi petrochimică.* Bucureşti, Editura didactică şi pedagogică, 1974.
8. Elmaghraby, S. E. *Proiectarea sistemelor de producţie.* Bucureşti, Editura tehnică, 1968.
9. Faure, R. *Eléments de la recherche opérationnelle.* Paris, Villars, 1971.
10. Feldbaum, A. *Principes théorique des systèmes asservis optimaux.* Moscou, MIR, 1973.
11. Forrester, G. W. *Principles of systems.* Cambridge, Massachusetts, 1968.
12. Forrester, G. W. *Industrial Dynamics.* Cambridge, Massachusetts, 1968.
13. Kaufmann, A. *Metode şi modele ale cercetării operaţionale,* vol. I şi II. Bucureşti, Editura ştiinţifică, 1967.
14. Kaufmann, A., Cruon R. *Les phénomènes d'attente.* Paris, Dunod, 1962.
15. Korn, A. G., Korn, M. Theresa. *Mathematical Handbook for Scientists and Engineers.* New York, McGraw-Hill Book, 1968.
16. Künzi, P. H., Krelle, W. *Nichtlineare Programmierung.* Berlin, Springer Verlag, 1962.
17. Maliţa, M., Zidăroiu, C. *Matematica organizării.* Bucureşti, Editura tehnică, 1971.
18. Marinoiu, V. *Teoria optimizării aplicată la reactoarele chimice* (rezumatul tezei de doctorat). Ploieşti, Litografia I.P.G., 1974.
19. Maruşciac, I. *Metode de rezolvare a problemelor de programare neliniară.* Cluj, Editura Dacia, 1973.
20. Maurin, H. *Programmation linéaire apliquée.* Paris, Editions Technip, 1967.
21. Mănescu, M., Dumitru, V., Ionescu, V., Bărbatu, I. Gh. *Programarea matematică în industria petrolieră.* Bucureşti, Editura Academiei Republicii Socialiste România, 1970.
22. Mihoc, Gh., Nădejde, I. *Programarea matematică,* vol. I şi II. Bucureşti, Editura ştiinţifică, 1966.

Scanned with OKEN Scanner

23. Rosenbrock, H. H., Storey, C. *Computational Techniques for Chemical Engineers.* London, Pergamon Press, 1966.
24. Savas, E. S. *Conducerea cu calculatoare a proceselor industriale.* Bucureşti, Editura tehnică, 1969.
25. Tamir, M. *Optimisation and simplification of system models charachterising an R & D process.* IEEE, 1972.
26. Teodorescu, N., Boldur, Gh., Stoica, M., Stancu-Minasian, I., Băncilă, I. *Metode ale cercetării operaţionale în gestiunea întreprinderilor.* Bucureşti, Editura tehnică, 1972.
27. Vajda, S. *Leçons sur la programmation mathématique.* Paris, Dunod, 1965.
28. Zadeh, L. A., Polak, E. ş.a. *Teoria sistemelor.* Bucureşti, Editura tehnică, 1973.
29. Zidăroiu, C. ş.a. *Probleme de cercetare operaţională.* Bucureşti, Editura Academiei Republicii Socialiste România, 1971.
30. Zidăroiu, C. *Programare dinamică discretă.* Bucureşti, Editura tehnică, 1975.
31. * * * Colecţie de programare pentru calculatorul electronic CET-500. Bucureşti, Editura Academiei Republicii Socialiste România, 1967.
32. CII. TRANSPOR — Ordinateur IRIS—50/*Programme de transport (manuel technique).* Louveciennes, CII, 1971 (R.I. S7.27).
33. CII. *Programmes de transport (manuel d'utilisation).* Louveciennes, CII, 1972 (4130 E/FR).
34. CII. OPALINE — *Programmation linéaire (manuel d'utilisation).* Louveciennes, CII, 1972 (4119 E/FR).
35. CII. *Programme de transport sous SIRIS 2 (manuel d'opérations).* Louveciennes, CII, 1972 (4133 E/FR).

Tiraj: 3350. Ex. S.P. 80 legate
Coli de tipar: 18. Planşe: 3
Hîrtia: Tipar înalt A. Format: 16/70×100
Bun de tipar: 11.X.1976. Ediţia: 1976
Nr. plan: 5475

Întreprinderea Poligrafică Cluj
Municipiul Cluj-Napoca
Str. Brassai Nr. 5—7
Republica Socialistă România
Comanda Nr. 235/1976

Scanned with OKEN Scanner